LINFOR3D User Manual

version 5.1.4

Matthias Steffen **Hans-Günter Ludwig** **Sven Wedemeyer-Böhm**

October 21, 2014

Contents

1 Introduction **6**

2 Getting started **6**

3 Basic Equations of Radiative Transfer **7**
3.1 Transfer equation for the continuum intensity . . . 7
3.2 Transfer equation for the line intensity . . . 8
3.3 Transfer equation for the line depression . . . 9
3.4 Contribution functions . . . 10
3.5 Grey test case . . . 12

4 Program Files and Data input files **17**
4.1 Main program flow . . . 17
4.2 Structures in Common Block linfordata . . . 18
4.3 IDL Files . . . 18

5 Parameter Input: linfor_setcmd.pro **20**
5.1 Program execution flags . . . 20
5.1.1 run_flag . . . 20
5.1.2 cv1_flag . . . 21
5.1.3 cv2_flag . . . 21
5.1.4 cv3_flag . . . 21
5.1.5 plt_flag . . . 21
5.1.6 maps_flag . . . 22
5.1.7 cc3d_flag . . . 22
5.1.8 nlte_flag . . . 22
5.1.9 free_flag . . . 23
5.2 General paths . . . 23
5.2.1 abupath . . . 23
5.2.2 ff_path . . . 23
5.2.3 opapath . . . 24
5.2.4 gaspath . . . 24
5.2.5 eospath . . . 24
5.3 Model data . . . 24
5.3.1 context . . . 24
5.3.2 rhdpath . . . 24
5.3.3 modelid . . . 25
5.3.4 parfs . . . 25
5.3.5 xbcpath . . . 25
5.3.6 abuid . . . 25
5.3.7 dmetal . . . 25
5.3.8 dalpha . . . 26
5.3.9 nx_skip . . . 26
5.3.10 ny_skip . . . 26
5.4 More model information (all read from parameter file for CO5BOLD data) . . . 26
5.4.1 opafile . . . 26
5.4.2 gasfile . . . 27
5.4.3 eosfile . . . 27
5.4.4 htau0 . . . 27
5.4.5 qmol . . . 27
5.4.6 Teff . . . 27

5.4.7 grav . . . 27
5.4.8 tsurffac . . . 27
5.5 Model data - reading of 'full' files (CO5BOLD only) . . . 28
5.5.1 isnap_full_1 . . . 28
5.5.2 isnap_full_2 . . . 28
5.5.3 istep_full . . . 28
5.6 $\langle 3D \rangle$ mean model . . . 29
5.6.1 mavg . . . 29
5.7 External 1D reference model . . . 29
5.7.1 atmpath . . . 29
5.7.2 atmfile . . . 29
5.8 Line data and radiative transfer . . . 29
5.8.1 linfs . . . 29
5.8.2 lutau1 . . . 30
5.8.3 lutau2 . . . 30
5.8.4 dlutau . . . 30
5.8.5 lctau1 . . . 30
5.8.6 lctau2 . . . 30
5.8.7 dlctau . . . 30
5.8.8 ntheta . . . 30
5.8.9 nphi . . . 31
5.8.10 mu0 . . . 31
5.8.11 kphi . . . 31
5.8.12 Hbrd . . . 31
5.8.13 ximicx . . . 32
5.8.14 ximic1 . . . 32
5.8.15 ximic3 . . . 32
5.8.16 ximacx . . . 32
5.8.17 ximac1 . . . 33
5.8.18 ximac3 . . . 33
5.8.19 vfacx . . . 33
5.8.20 vfacy . . . 33
5.8.21 vfacz . . . 33
5.8.22 micro . . . 33
5.8.23 xi_a . . . 34
5.8.24 xi_b . . . 34
5.8.25 xi_d . . . 34
5.8.26 dclam . . . 34
5.8.27 intmode . . . 34
5.8.28 intline . . . 35
5.8.29 nchunk . . . 35
5.9 Example . . . 36

6 Line Data File: line.dat **38**
6.1 Parameters in Line Data File . . . 38
6.1.1 clam . . . 38
6.1.2 gfscale . . . 38
6.2 Line Data Formats . . . 38
6.2.1 **Continuum only** . . . 38
6.2.2 **Single line calculations, line data format '0'** . . . 39
6.2.3 **Single line calculations, line data format '1'** . . . 40
6.2.4 **Single line calculations, complete line data format '2'** . . . 42

6.2.5 **Single line calculations, complete line data format '3'** 43
6.2.6 **Single line calculations, complete line data format '7'** 44
6.2.7 **Multiple Line Calculations** 44
6.3 **Conversion of line broadening parameters** 45
6.3.1 Quadratic Stark effect 45
6.3.2 Van der Waals broadening 46
6.3.3 Natural line broadening 47

7 Output files **48**
7.1 Files containing intensity maps 48
7.2 Foreshortening 49

8 Timing statistics **50**

9 IONDIS **51**
9.1 Molecules 51
9.1.1 Some definitions 51
9.1.2 Equations 51
9.1.3 Criterion for convergence 54
9.1.4 Initial guess 54
9.1.5 Variable names 55

List of Figures

1 Analytical curve-of-growth 14

List of Tables

1 List of all structures in common blocks 'linfordata' 18
2 List of additional IDL modules 18
3 List of all IDL modules 19
4 Small molecular network: 5 atoms, 8 molecules 51
5 Large molecular network:10 atoms, 14 molecules 51

1 Introduction

LINFOR3D is based on the old code `LINFOR`. The present version of LINFOR3D is still preliminary. The aim is to develop the code in IDL and to transform it to FORTRAN for better execution efficiency, after it is checked and found to work properly under IDL.
Hence, users should be aware of the following limitations:

- **Geometry:** This version is limited to spectrum synthesis from **local hydrodynamical models** (solar-type convective atmospheres). In a future version, it should be possible to compute the line formation in **global giant models**.
- **Efficiency:** No effort was yet taken to really improve the execution speed of this IDL/FORTRAN code. In fact there are still some parts in the code which are unnecessary for the current operation. Their execution should be controlled by additional flags in a future release.

Like the code itself this user manual is under construction, too. Nevertheless, it will be of substantial help while creating new command and spectral line data files for LINFOR3D.
To get a brief overview you might want to look into the section "Getting started" (Sect. 2) first.

2 Getting started

First make sure that you have all files which are listed in Tab. 3 and 2. These files should be put together in a directory which can be accessed under IDL.

Now two files have to be edited and provided in order to run LINFOR3D:

- `linforset_cmd.pro`:
  This file, which is an IDL script, defines the data structure `cmd`. This structure contains all necessary information (except for spectral line data) like, e.g., paths and names of model file(s). See Sect. 5 for more details.
- `line.dat`: This file contains all data for spectral line such as, e.g., oscillator strength and broadening parameters. The usual file name is `line.dat` but it might be given another name which then has be to entered in `linfor_setcmd.pro`. See Sect. 6 for more details.

After done so, you might start IDL and start LINFOR3D by simply typing

```
IDL> .r linfor_3D.pro
```

Several output files are created. It is also possible to directly use the results under IDL. See Sect. 7 for more details.
Note that LINFOR3D stores the flow field in temporary cache files which are automatically restored if the same calculation is repeated!

3 Basic Equations of Radiative Transfer

3.1 Transfer equation for the continuum intensity

$$\frac{\mathrm{d}\, I^c_\lambda}{\mathrm{d}\, s} = -\kappa^c_\lambda\, I^c_\lambda + \kappa^c_\lambda\, S^c_\lambda \tag{1}$$

together with the definition of the optical depth along the ray

$$\mathrm{d}\,\tau^c_\lambda = -\kappa^c_\lambda\, \mathrm{d}\, s\,, \tag{2}$$

reads

$$\frac{\mathrm{d}\, I^c_\lambda}{\mathrm{d}\,\tau^c_\lambda} = I^c_\lambda - S^c_\lambda\,. \tag{3}$$

The solution of Eq. (3) is

$$I^c_\lambda(\tau^c_\lambda) = \int_{\tau^c_\lambda}^{\tau^b_\lambda} S^c_\lambda(\tau')\, \exp\{-(\tau' - \tau^c_\lambda)\}\, \mathrm{d}\,\tau' \;+\; I^c_\lambda(\tau^b_\lambda)\, \exp\{-(\tau^b_\lambda - \tau^c_\lambda)\} \tag{4}$$

where τ^b_λ is the continuum optical depth at the lower boundary. The **emergent** continuum intensity is:

$$I^c_\lambda(\tau^c_\lambda = 0) = \int_0^{\tau^b_\lambda} S^c_\lambda(\tau')\, \exp\{-\tau'\}\, \mathrm{d}\,\tau' \;+\; I^c_\lambda(\tau^b_\lambda)\, \exp\{-\tau^b_\lambda\}\,. \tag{5}$$

Defining

$$u^c_\lambda = I^c_\lambda - S^c_\lambda \tag{6}$$

we have the transport equation

$$\frac{\mathrm{d}\, u^c_\lambda}{\mathrm{d}\,\tau^c_\lambda} = u^c_\lambda - \frac{\mathrm{d}\, S^c_\lambda}{\mathrm{d}\,\tau^c_\lambda}\,. \tag{7}$$

The solution for u^c_λ is found by replacing S^c_λ by $\mathrm{d}\, S^c_\lambda/\mathrm{d}\,\tau^c_\lambda$ in Eq.(4):

$$u^c_\lambda(\tau^c_\lambda) = \int_{\tau^c_\lambda}^{\tau^b_\lambda} \frac{\mathrm{d}\, S^c_\lambda(\tau')}{\mathrm{d}\,\tau^c_\lambda}\, \exp\{-(\tau' - \tau^c_\lambda)\}\, \mathrm{d}\,\tau' \;+\; u^c_\lambda(\tau^b_\lambda)\, \exp\{-(\tau^b_\lambda - \tau^c_\lambda)\} \tag{8}$$

The emergent intensity can also be obtained from Eq.(8):

$$I^c_\lambda(\tau^c_\lambda = 0) = S^c_\lambda(\tau^c_\lambda = 0) + \int_0^{\tau^b_\lambda} \frac{\mathrm{d}\, S^c_\lambda(\tau')}{\mathrm{d}\,\tau^c_\lambda}\, \exp\{-\tau'\}\, \mathrm{d}\,\tau' \;+\; u^c_\lambda(\tau^b_\lambda)\, \exp\{-\tau^b_\lambda\}\,. \tag{9}$$

Now we define a fixed central wavelength, λ_0, with the corresponding **fixed (universal) optical depth scale** τ_0, which is equidistant in $\log\tau_0$ and may used alternatively for all integrations. On this optical depth scale, Eq.(4) becomes

$$I^c_\lambda(\tau_0) = \int_{\tau_0}^{\tau^b_0} \frac{\kappa^c_\lambda}{\kappa^c_0}\, S^c_\lambda(\tau'_0)\, \exp\{-\left(\tau^c_\lambda(\tau'_0) - \tau^c_\lambda(\tau_0)\right)\}\, \mathrm{d}\,\tau'_0 \;+\; I^c_\lambda(\tau^b_0)\, \exp\{-\left(\tau^c_\lambda(\tau^b_0) - \tau^c_\lambda(\tau_0)\right)\}\,, \tag{10}$$

giving the continuum intensity at wavelength λ as a function of optical depth τ_0. Note the factor $\kappa^c_\lambda/\kappa^c_0$ under the integral. The intensity at the lower boundary, $I^c_\lambda(\tau^b_0)$, can be computed from the diffusion approximation,

$$I^c_\lambda(\tau^b_0) = S^c_\lambda(\tau^b_0) + \frac{\kappa^c_0}{\kappa^c_\lambda}\, \frac{\mathrm{d}\, S^c_\lambda}{\mathrm{d}\,\tau_0}(\tau^b_0)\,, \tag{11}$$

but the boundary term may also be neglected, at least for the emergent intensity, because the exponential factor is usually very small. For the **emergent** intensity we have from Eq.(5):

$$I^c_\lambda(\tau_0 = 0) = \int_0^{\tau^b_0} \frac{\kappa^c_\lambda}{\kappa^c_0}\, S^c_\lambda(\tau'_0)\, \exp\{-\tau^c_\lambda(\tau'_0)\}\, \mathrm{d}\,\tau'_0 \;+\; I^c_\lambda(\tau^b_0)\, \exp\{-\tau^c_\lambda(\tau^b_0)\}\,. \tag{12}$$

Similarly, Eq.(8) becomes

$$u_\lambda^c(\tau_0) = \int_{\tau_0}^{\tau_0^b} \frac{\mathrm{d}\, S_\lambda^c}{\mathrm{d}\,\tau_0}(\tau_0') \, \exp\{-\tau_\lambda^c(\tau_0') - \tau_\lambda^c(\tau_0)\} \, \mathrm{d}\,\tau_0' \; + \; u_\lambda^c(\tau_0^b) \, \exp\{-\tau_\lambda^c(\tau_0^b) - \tau_\lambda^c(\tau_0)\} \, . \tag{13}$$

Note the absence of the factor $\kappa_\lambda^c/\kappa_0^c$ under the integral in this case. $u_\lambda^c(\tau_0^b)$ is obtained from the diffusion approximation,

$$u_\lambda^c(\tau_0^b) = \frac{\kappa_0^c}{\kappa_\lambda^c} \, \frac{\mathrm{d}\, S_\lambda^c}{\mathrm{d}\,\tau_0}(\tau_0^b). \tag{14}$$

The emergent intensity can be computed from Eq.(13) as:

$$I_\lambda^c(\tau_0 = 0) = S_\lambda^c(\tau_0 = 0) + \int_0^{\tau_0^b} \frac{\mathrm{d}\, S_\lambda^c}{\mathrm{d}\,\tau_0}(\tau_0') \, \exp\{-\tau_\lambda^c(\tau_0')\} \, \mathrm{d}\,\tau_0' \; + \; u_\lambda^c(\tau_0^b) \, \exp\{-\tau_\lambda^c(\tau_0^b)\} \, . \tag{15}$$

In the latest version of Linfor3D, the continuum intensity is calculated from Eqs.(8) and (9), at 3 different wavelengths: $\lambda_0 - \Delta\lambda$, λ_0, and $\lambda_0 + \Delta\lambda$, where $\Delta\lambda$ is specified by the parameter `dclam`. We ensure that the derivative $\mathrm{d}\, S_\lambda^c/\mathrm{d}\,\tau_0$ fulfills the condition

$$\int_{\tau_1}^{\tau_2} \frac{\mathrm{d}\, S_\lambda^c}{\mathrm{d}\,\tau_\lambda}(\tau_\lambda') \, \mathrm{d}\,\tau_\lambda' = S_\lambda^c(\tau_2) - S_\lambda^c(\tau_1) \, . \tag{16}$$

The reason for using Eqs.(8) and (9) instead of Eq.(5) is that the quantity $u_\lambda^c(\tau)$ is needed to compute the line depression source function (see Sect. 3.3). We have checked that the usual transfer equation, Eq.(5), gives numerically very closely the same results for the emergent continuum intensity as Eq.(9).

3.2 Transfer equation for the line intensity

In the presence of lines, the transfer equation at wavelength λ reads

$$\frac{\mathrm{d}\, I_\lambda^\ell}{\mathrm{d}\, s} = -\left(\kappa_\lambda^c + \sum_\ell \kappa_\lambda^\ell\right) I_\lambda^\ell + \kappa_\lambda^c \, S_\lambda^c + \sum_\ell \kappa_\lambda^\ell \, S_\lambda^\ell \, . \tag{17}$$

The line source functions S_λ^ℓ may be different from the LTE continuum source function S_λ^c. With the definition of the total optical depth

$$\mathrm{d}\,\tau_\lambda = -\left(\kappa_\lambda^c + \sum_\ell \kappa_\lambda^\ell\right) \mathrm{d}\, s \equiv \mathrm{d}\,\tau_\lambda^c + \mathrm{d}\,\tau_\lambda^\ell \, , \tag{18}$$

and the total source function

$$S_\lambda = \frac{\kappa_\lambda^c \, S_\lambda^c}{\kappa_\lambda^c + \sum_\ell \kappa_\lambda^\ell} + \frac{\sum_\ell \kappa_\lambda^\ell S_\lambda^\ell}{\kappa_\lambda^c + \sum_\ell \kappa_\lambda^\ell} = \frac{S_\lambda^c + \eta \, \overline{S_\lambda^\ell}}{1 + \eta} = \frac{1 + \beta}{1 + \eta} \, S_\lambda^c \, , \tag{19}$$

where

$$\overline{S_\lambda^\ell} = \frac{\sum_\ell \kappa_\lambda^\ell S_\lambda^\ell}{\sum_\ell \kappa_\lambda^\ell} \, , \quad \eta = \frac{\sum_\ell \kappa_\lambda^\ell}{\kappa_\lambda^c} \, , \quad \beta = \frac{\sum_\ell \kappa_\lambda^\ell S_\lambda^\ell}{\kappa_\lambda^c \, S_\lambda^c} \, , \tag{20}$$

we can write

$$\frac{\mathrm{d}\, I_\lambda^\ell}{\mathrm{d}\,\tau_\lambda} = I_\lambda^\ell - S_\lambda \, . \tag{21}$$

In LTE, $S_\lambda = S_\lambda^c$. The solution of Eq.(21) is

$$I_\lambda^\ell(\tau_\lambda = 0) = \int_0^{\tau_\lambda^b} S_\lambda(\tau_\lambda') \, \exp\{-\tau_\lambda'\} \, \mathrm{d}\,\tau_\lambda' \; + \; I_\lambda^\ell(\tau_\lambda^b) \, \exp\{-\tau_\lambda^b\} \, . \tag{22}$$

In analogy to Eq.(12), we can also obtain the emergent line intensity by integration on the universal optical depth scale τ_0:

$$I_\lambda^\ell(\tau_0 = 0) = \int_0^{\tau_0^b} \frac{\kappa_\lambda^c}{\kappa_0^c}\,(1+\eta)\,S_\lambda(\tau_0')\,\exp\{-\tau_\lambda(\tau_0')\}\,\mathrm{d}\,\tau_0' \;+\; I_\lambda^\ell(\tau_0^b)\,\exp\{-\tau_\lambda(\tau_0^b)\}\,, \tag{23}$$

or, substituting S_λ from Eq.(19),

$$I_\lambda^\ell(\tau_0 = 0) = \int_0^{\tau_0^b} \frac{\kappa_\lambda^c}{\kappa_0^c}\,(1+\beta)\,S_\lambda^c(\tau_0')\,\exp\{-\tau_\lambda(\tau_0')\}\,\mathrm{d}\,\tau_0' \;+\; I_\lambda^\ell(\tau_0^b)\,\exp\{-\tau_\lambda(\tau_0^b)\}\,. \tag{24}$$

Integration on the $\log\tau_0$ scale ($z_0 \equiv \log\tau_0$) gives:

$$I_\lambda^\ell(z_0^a) = \int_{z_0^a}^{z_0^b} \ln(10)\,\tau_0(z_0')\,\frac{\kappa_\lambda^c}{\kappa_0^c}\,(1+\beta)\,S_\lambda^c(z_0')\,\exp\{-\tau_\lambda(z_0')\}\,\mathrm{d}\,z_0' \;+\; I_\lambda^\ell(z_0^b)\,\exp\{-\tau_\lambda(z_0^b)\}\,, \tag{25}$$

where z_0^a is the minimum log optical depth. Alternatively, in analogy to Eq.(9) we obtain:

$$I_\lambda^\ell(\tau_\lambda = 0) = S_\lambda(\tau_\lambda = 0) + \int_0^{\tau_\lambda^b} \frac{\mathrm{d}\,S_\lambda}{\mathrm{d}\,\tau_\lambda}(\tau_\lambda')\,\exp\{-\tau_\lambda'\}\,\mathrm{d}\,\tau_\lambda' \;+\; u_\lambda^\ell(\tau_\lambda^b)\,\exp\{-\tau_\lambda^b\}\,, \tag{26}$$

where we have defined

$$u_\lambda^\ell = I_\lambda^\ell - S_\lambda\,, \tag{27}$$

which in the diffusion approximation may be written as

$$u_\lambda^\ell(\tau_\lambda^b) = \frac{\mathrm{d}\,S_\lambda}{\mathrm{d}\,\tau_\lambda}(\tau_\lambda^b) \quad \text{or} \quad u_\lambda^\ell(\tau_0^b) = \frac{\kappa_0^c/\kappa_\lambda^c}{1+\eta}\,\frac{\mathrm{d}\,S_\lambda}{\mathrm{d}\,\tau_0}(\tau_0^b). \tag{28}$$

On the universal optical depth scale τ_0 we obtain from Eq.(26:

$$I_\lambda^\ell(\tau_0 = 0) = S_\lambda(\tau_0 = 0) + \int_0^{\tau_0^b} \frac{\mathrm{d}\,S_\lambda}{\mathrm{d}\,\tau_0}(\tau_0')\,\exp\{-\tau_\lambda(\tau_0')\}\,\mathrm{d}\,\tau_0' \;+\; u_\lambda^\ell(\tau_0^b)\,\exp\{-\tau_\lambda(\tau_0^b)\}\,, \tag{29}$$

In LTE, where $S_\lambda = S_\lambda^c$, the integral in Eq.(29) differs from the integral in Eq.(15) only by the exponential factor which involves the total optical depth τ_λ instead of the continuum optical depth τ_λ^c. The absolute line depression is then calculated as

$$D_\lambda = I_\lambda^c(\tau = 0) - I_\lambda^\ell(\tau = 0)\,. \tag{30}$$

In the current version of Linfor3D, Eq.(25) is used if the parameter `intline` is set to -1, and Eq.(26) is used if `intline` is set to -2.

3.3 Transfer equation for the line depression

We may analyse the transfer equation for the absolute line depression defined in Eq.(30):

$$\frac{\mathrm{d}\,D_\lambda}{\mathrm{d}\,s} = \frac{\mathrm{d}\,I_\lambda^c}{\mathrm{d}\,s} - \frac{\mathrm{d}\,I_\lambda^\ell}{\mathrm{d}\,s} = -\kappa_\lambda^c\,I_\lambda^c + \left(\kappa_\lambda^c + \sum_\ell \kappa_\lambda^\ell\right)\,I_\lambda^\ell - \sum_\ell \kappa_\lambda^\ell\,S_\lambda^\ell \tag{31}$$

or

$$\frac{\mathrm{d}\,D_\lambda}{\mathrm{d}\,s} = -\left(\kappa_\lambda^c + \sum_\ell \kappa_\lambda^\ell\right)\,D_\lambda + \left(I_\lambda^c \sum_\ell\,\kappa_\lambda^\ell\; - \sum_\ell\,\kappa_\lambda^\ell\,S_\lambda^\ell\right) \tag{32}$$

or

$$\frac{\mathrm{d}\,D_\lambda}{\mathrm{d}\,\tau_\lambda} = D_\lambda - S_\lambda^D\,, \tag{33}$$

where the line depression source function is

$$S_\lambda^D = \frac{\eta}{1+\eta}\left(I_\lambda^c - \overline{S_\lambda^\ell}\right) = \frac{\eta}{1+\eta}\left((I_\lambda^c - S_\lambda^c) + (S_\lambda^c - \overline{S_\lambda^\ell})\right). \tag{34}$$

In LTE, $\overline{S_\lambda^\ell} = S_\lambda^c$, and

$$S_\lambda^D = \frac{\eta}{1+\eta}\,(I_\lambda^c - S_\lambda^c)\,. \tag{35}$$

The solution of Eq.(33) is

$$D_\lambda(\tau_\lambda = 0) = \int_0^{\tau_\lambda^b} S_\lambda^D(\tau'_\lambda)\,\exp\{-\tau'_\lambda\}\,\mathrm{d}\,\tau'_\lambda\,, \tag{36}$$

neglecting the boundary term. Integration on the fixed τ_0 scale:

$$D_\lambda(\tau_0 = 0) = \int_0^{\tau_0^b} \frac{\kappa_\lambda^c}{\kappa_0^c}\,(1+\eta)\,S_\lambda^D(\tau'_0)\,\exp\{-\tau_\lambda(\tau'_0)\}\,\mathrm{d}\,\tau'_0\,. \tag{37}$$

Substituting S_λ^D from Eq.(34) gives

$$D_\lambda(\tau_0 = 0) = \int_0^{\tau_0^b} \frac{\kappa_\lambda^c}{\kappa_0^c}\,\eta\,\left(I_\lambda^c - \overline{S_\lambda^\ell}\right)\,\exp\{-\tau_\lambda(\tau'_0)\}\,\mathrm{d}\,\tau'_0\,, \tag{38}$$

where κ_λ^c, κ_0^c, η, I_λ^c, $\overline{S_\lambda^\ell}$, and τ_λ are defined as a function of τ_0. We can also write

$$D_\lambda(\tau_0 = 0) = \int_0^{\tau_0^b} \frac{\kappa_\lambda^c}{\kappa_0^c}\,\eta\,\left(u_\lambda^c + (S_\lambda^c - \overline{S_\lambda^\ell})\right)\,\exp\{-\tau_\lambda(\tau'_0)\}\,\mathrm{d}\,\tau'_0\,, \tag{39}$$

where

$$\frac{\eta}{1+\eta}\,(S_\lambda^c - \overline{S_\lambda^\ell}) = S_\lambda^c\,\frac{(\eta-\beta)}{1+\eta} \tag{40}$$

is the NLTE correction to the line depression source function. Integration on the $\log\tau_0$ scale ($z_0 \equiv \log\tau_0$) gives:

$$D_\lambda(z_0^a) = \int_{z_0^a}^{z_0^b} \ln(10)\,\tau_0(z'_0)\,\frac{\kappa_\lambda^c}{\kappa_0^c}\,\eta\,\left(u_\lambda^c + (S_\lambda^c - \overline{S_\lambda^\ell})\right)\,\exp\{-\tau_\lambda(z'_0)\}\,\mathrm{d}\,z'_0\,. \tag{41}$$

In the current version of Linfor3D, Eq.(41) is used to compute the line depression if the parameter `intline` is set to 1, while Eq.(36) is used if `intline` = 2.

3.4 Contribution functions

The *Continuum Intensity Contribution Function* for a ray with inclination angle $\mu = \cos\theta$, azimuthal angle ϕ, and wavelength λ is simply the horizontal average of the integrand of Eq.(12)

$$C_I^c(\tau_0, \mu, \phi, \lambda) = \frac{1}{\mu}\left\langle \frac{\kappa_\lambda^c}{\kappa_0^c}\,S_\lambda^c(\tau_0/\mu)\,\exp\{-\tau_\lambda^c(\tau_0/\mu)\}\right\rangle_{x,y}, \tag{42}$$

such that

$$I_\lambda^c(\tau_0 = 0, \mu, \phi, \lambda) = \int_0^{\tau_0^b} C_I^c(\tau'_0, \mu, \phi, \lambda)\,\mathrm{d}\,\tau'_0 = \int_0^{z_0^b} \ln(10)\,\tau_0(z'_0)\,C_I^c(\tau_0(z'_0), \mu, \phi, \lambda)\,\mathrm{d}\,z'_0\,. \tag{43}$$

Note that now (τ_0/μ) is the optical depth along the line-of-sight, and τ_0 is the corresponding vertical optical depth (a *formal* quantity in the presence of horizontal inhomogeneities).

The *Continuum Flux Contribution Function* at wavelength λ is consequently

$$C_F^c(\tau_0,\lambda) = \int_0^{2\pi}\int_0^1 \mu'\, C_I^c(\tau_0,\mu',\phi',\lambda)\,\mathrm{d}\,\mu'\,\mathrm{d}\,\phi'\,, \tag{44}$$

such that

$$F_\lambda^c(\tau_0=0,\lambda) = \int_0^{\tau_0^b} C_F^c(\tau_0',\lambda)\,\mathrm{d}\,\tau_0' \;= \int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,C_F^c(\tau_0(z_0'),\lambda)\,\mathrm{d}\,z_0'\,. \tag{45}$$

Note that the horizontal averaging in Eq.(42) works only because the transfer equation is integrated on the fixed universal optical depth scale, τ_0. The contribution functions $C_I^c(\tau_0,\mu_0,\phi_0,\lambda_0)$ and $C_F^c(\tau_0,\lambda_0)$ are saved in `contf.cfc3i` and `contf.cfc3f`, respectively. Corresponding contribution functions are also computed for the $\langle 3D\rangle$ model and saved in `contf.cfc1i` and `contf.cfc1f`, respectively, and for the external 1D reference atmosphere (`contf.cfcxi` and `contf.cfcxf`).

Similarly, we can also write down the *Line Intensity Contribution Function* as the horizontal average of the integrand of Eq.(24):

$$C_I^\ell(\tau_0,\mu,\phi,\lambda) = \frac{1}{\mu}\left\langle \frac{\kappa_\lambda^c}{\kappa_0^c}\,(1+\beta)\,S_\lambda^c(\tau_0/\mu)\,\exp\{-\tau_\lambda(\tau_0/\mu)\}\right\rangle_{x,y}\,, \tag{46}$$

such that the intensity at a given wavelength in the line profile is

$$I_\lambda^\ell(\tau_0=0,\mu,\phi,\lambda) = \int_0^{\tau_0^b} C_I^\ell(\tau_0',\mu,\phi,\lambda)\,\mathrm{d}\,\tau_0' \;= \int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,C_I^\ell(\tau_0(z_0'),\mu,\phi,\lambda)\,\mathrm{d}\,z_0'\,. \tag{47}$$

The *Line Flux Contribution Function* at wavelength λ is

$$C_F^\ell(\tau_0,\lambda) = \int_0^{2\pi}\int_0^1 \mu'\, C_I^\ell(\tau_0,\mu',\phi',\lambda)\,\mathrm{d}\,\mu'\,\mathrm{d}\,\phi'\,, \tag{48}$$

such that

$$F_\lambda^\ell(\tau_0=0,\lambda) = \int_0^{\tau_0^b} C_F^\ell(\tau_0',\lambda)\,\mathrm{d}\,\tau_0' \;= \int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,C_F^\ell(\tau_0(z_0'),\lambda)\,\mathrm{d}\,z_0'\,. \tag{49}$$

$C_I^\ell(\tau_0,\mu_0,\phi_0,\lambda_0)$ and $C_F^\ell(\tau_0,\lambda_0)$ are stored in `contf.cfl3i` and `contf.cfl3f`, respectively, and similarly for the 1D atmospheres in `contf.cfl1i`, `contf.cfl1f`, `contf.cflxi`, and `contf.cflxf`.

Formally, a *Line Depression Contribution Function* could be defined as

$$\tilde{C}_I^D = C_I^c - C_I^\ell = \frac{1}{\mu}\left\langle \frac{\kappa_\lambda^c}{\kappa_0^c}\,S_\lambda^c(\tau_0/\mu)\,\exp\{-\tau_\lambda^c(\tau_0/\mu)\}\left(1-(1+\beta)\,\exp\{-\tau_\lambda^\ell(\tau_0/\mu)\}\right)\right\rangle_{x,y}\,, \tag{50}$$

such that the absolute line depression at any wavelength in the line profile is

$$D_I(\tau_0=0,\mu,\phi,\lambda) = \int_0^{\tau_0^b} \tilde{C}_I^D(\tau_0',\mu,\phi,\lambda)\,\mathrm{d}\,\tau_0' \;= \int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,\tilde{C}_I^D(\tau_0(z_0'),\mu,\phi,\lambda)\,\mathrm{d}\,z_0'\,. \tag{51}$$

Note however, that $\tilde{C}_I^D$ does not have the desired physical meaning, because the factor $(1-(1+\beta)\,\exp\{-\tau_\lambda^\ell\})$ (i) becomes negative when τ_λ^ℓ is small (τ_λ^ℓ is the optical depth due to the line opacity only), and (ii) it is non-zero also in layers where the line opacity vanishes. For this reason, $\tilde{C}_I^D$ is not considered useful and hence is not computed in the current version of LINFOR3D.

A much better way to define the *Line Depression Contribution Function* is to consider Eq.(39) and to define it as

$$C_I^D(\tau_0,\mu,\phi,\lambda) = \frac{1}{\mu}\left\langle \frac{\kappa_\lambda^c}{\kappa_0^c}\,(\eta\,u_\lambda^c(\tau_0/\mu) + (\eta-\beta)\,S_\lambda^c(\tau_0/\mu))\,\exp\{-\tau_\lambda(\tau_0/\mu)\}\right\rangle_{x,y}\,. \tag{52}$$

Note that this contribution function vanisches whereever the line opacity (η, β) is zero. For the flux spectrum we define, as before,

$$C_F^D(\tau_0,\lambda) = \int_0^{2\pi}\int_0^1 \mu'\, C_I^D(\tau_0,\mu',\phi',\lambda)\,\mathrm{d}\,\mu'\,\mathrm{d}\,\phi'\,. \tag{53}$$

Then the absolute line depression at any wavelength in the line profile is

$$D_I(\tau_0=0,\mu,\phi,\lambda) = \int_0^{\tau_0^b} C_I^D(\tau_0',\mu,\phi,\lambda)\,\mathrm{d}\,\tau_0' = \int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,C_I^D(\tau_0(z_0'),\mu,\phi,\lambda)\,\mathrm{d}\,z_0'\,, \tag{54}$$

and

$$D_F(\tau_0=0,\lambda) = \int_0^{\tau_0^b} C_F^D(\tau_0',\lambda)\,\mathrm{d}\,\tau_0' = \int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,C_F^D(\tau_0(z_0'),\lambda)\,\mathrm{d}\,z_0'\,, \tag{55}$$

for the intensity and flux spectrum, respectively.

The*Equivalent Width Contribution Function* is computed as

$$C_I^W(\tau_0,\mu,\phi) = \int_\lambda C_I^D(\tau_0,\mu,\phi,\lambda')\mathrm{d}\,\lambda'\,, \tag{56}$$

and

$$\begin{aligned} W_I(\mu,\phi) &= \frac{1}{\langle I_\lambda^c(\mu,\phi,\lambda)\rangle}\int_0^{\tau_0^b} C_I^W(\tau_0',\mu,\phi)\,\mathrm{d}\,\tau_0' \\ &= \frac{1}{\langle I_\lambda^c(\mu,\phi,\lambda)\rangle}\int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,C_I^W(\tau_0(z_0'),\mu,\phi)\,\mathrm{d}\,z_0'\,, \end{aligned} \tag{57}$$

where

$$\langle I_\lambda^c(\mu,\phi,\lambda)\rangle = \frac{\int_\lambda D_I(\mu,\phi,\lambda')\,\mathrm{d}\,\lambda'}{\int_\lambda D_I(\mu,\phi,\lambda')/I_\lambda^c(\mu,\phi,\lambda')\,\mathrm{d}\,\lambda'}\,. \tag{58}$$

For the flux spectrum we have

$$C_F^W(\tau_0) = \int_\lambda C_F^D(\tau_0,\lambda')\mathrm{d}\,\lambda'\,, \tag{59}$$

and

$$\begin{aligned} W_F &= \frac{1}{\langle F_\lambda^c(\lambda)\rangle}\int_0^{\tau_0^b} C_F^W(\tau_0')\,\mathrm{d}\,\tau_0' \\ &= \frac{1}{\langle F_\lambda^c(\lambda)\rangle}\int_0^{z_0^b} \ln(10)\,\tau_0(z_0')\,C_F^W(\tau_0(z_0'))\,\mathrm{d}\,z_0'\,, \end{aligned} \tag{60}$$

with

$$\langle F_\lambda^c(\lambda)\rangle = \frac{\int_\lambda D_F(\lambda')\,\mathrm{d}\,\lambda'}{\int_\lambda D_F(\lambda')/F_\lambda^c(\lambda')\,\mathrm{d}\,\lambda'}\,. \tag{61}$$

Irrespective of the parameter `intline`, the structures `contf.cfd3i` and `contf.cfd3f` hold the contribution functions $C_I^D(\tau_0,\mu 0,\phi_0,\lambda 0)$ and $C_F^D(\tau_0,\lambda 0)$, while $C_I^W(\tau_0,\mu 0,\phi_0)$ and $C_F^W(\tau_0)$ are stored in `contf.cfw3i` and `contf.cfw3f`, respectively.

3.5 Grey test case

If `cmd.context` is set to 'grey', a 3D ($n_x = n_y = 10$) hydrostatic atmosphere is constructed, instead of reading a 3D model. The temperature stratification on the Rosseland optical depth scale is given by

$$T(\tau_{\mathrm{Ross}}) = T_{\mathrm{eff}}\left(\frac{1}{2}+\frac{3}{4}\,\tau_{\mathrm{Ross}}\right)^{1/4} \tag{62}$$

and the source function is linear in $\tau_{\rm Ross}$:

$$S(\tau_{\rm Ross}) = \frac{\sigma}{\pi}\, T_{\rm eff}^4 \, \left(\frac{1}{2} + \frac{3}{4}\, \tau_{\rm Ross}\right). \tag{63}$$

The Eddington-Barbier relation is strictly correct in this case. For any inclination $\mu = \cos\,\theta$, the emergent continuum intensity is given by

$$I_c(\mu) = \frac{\sigma}{\pi}\, T_{\rm eff}^4 \, \left(\frac{1}{2} + \frac{3}{4}\, \mu\right). \tag{64}$$

In particular, at disk-center ($\mu = 1$) the continuum intensity is

$$I_c(\mu = 1) = \frac{5}{4}\, \frac{\sigma}{\pi}\, T_{\rm eff}^4, \tag{65}$$

and the flux is

$$F_c = 2\,\pi \int_0^1 \mu\, I_c(\mu)\, {\rm d}\,\mu = \sigma\, T_{\rm eff}^4. \tag{66}$$

Comparison of the results obtained from LINFOR3D for continuum intensity and flux for

```
TEFF = 5000.00, GRAV = 316.200
LUTAU1 = -8.0000000, LUTAU2 = 2.0000000, DLUTAU = 0.0800000
OPAFILE = 't5000g250mm30_marcs_idmean3xRT3.opta',
GASFILE = 'gas_cifist2006_m30_a04_l5.eos',
EOSFILE = 'eos_cifist2006_m30_a04_l5.eos'
```

with the above theoretical results yields (LINFOR3D 3.1.3):

`ratio`	`ntheta`				
numerical / analytical	1	2	3	−3	4
I_c(linfor3D)/I_c(Eq.(65)	1.0005573	1.0005573	1.0005573	1.0005573	1.0005573
F_c(linfor3D)/F_c(Eq.(66)	1.0004105	1.0148776	1.0079553	1.0004507	1.0050481

If the ratio η of line opacity, κ_ℓ and continuum opacity, κ_c is constant with optical depth ($\eta = \kappa_\ell/\kappa_c$), the intensity in the line is simply

$$I_\ell(\mu) = \frac{\sigma}{\pi}\, T_{\rm eff}^4 \, \left(\frac{1}{2} + \frac{3}{4}\, \frac{\mu}{1+\eta}\right), \tag{67}$$

the **absolute** line depression is

$$D_I(\mu) = I_c(\mu) - I_\ell(\mu) = \frac{\sigma}{\pi}\, T_{\rm eff}^4\, \frac{3}{4}\, \mu\, \frac{\eta}{1+\eta}, \tag{68}$$

and the **relative** line depression at disk-center is

$$D_I(\mu = 1)/I_c(\mu = 1) = \frac{3}{5}\, \frac{\eta}{1+\eta}. \tag{69}$$

The absolute line depression for flux is

$$D_F = F_c - F_\ell = 2\,\pi \int_0^1 \mu\, D_I(\mu)\, {\rm d}\,\mu = \sigma\, T_{\rm eff}^4\, \frac{1}{2}\, \frac{\eta}{1+\eta}, \tag{70}$$

and the relative line depression for flux is

$$D_F/F_c = \frac{1}{2}\, \frac{\eta}{1+\eta}. \tag{71}$$

The ratio between the relative line depression in flux and at disk-center is therfore 5/6, and the same ratio holds for the equivalent widths.

The local absorption line profile is now defined by

$$\eta(\alpha, v) = \eta_0 \, H(\alpha, v) \,, \tag{72}$$

where $v = (\lambda - \lambda_0)/\Delta\lambda_D$, and $\alpha = \Delta\lambda_N/2/\Delta\lambda_D$ ($\Delta\lambda_D$: Doppler width, $\Delta\lambda_N$: full width at half maximum of the Lorentzian damping profile). The 'Voigt function' $H(\alpha, v)$ is normalized such that (for $\alpha \ll 1$), $H(\alpha, v = 0) \approx 1$. Assuming that η_0, α, and $\Delta\lambda_D$ are constant, we can compute the emergent line profile from Eq. (69) or (71). At disk-center, we have

$$D_I(\mu = 1)/I_c(\mu = 1) = R_I = \frac{3}{5} \, \frac{\eta_0 \, H(\alpha, v)}{\eta_0 \, H(\alpha, v) + 1} \,, \tag{73}$$

and for flux

$$D_F/F_c = R_F = \frac{1}{2} \, \frac{\eta_0 \, H(\alpha, v)}{\eta_0 \, H(\alpha, v) + 1} \,. \tag{74}$$

Clearly, the emergent line profiles are no longer Voigt profiles due to saturation effects.

The (reduced) disk-center equivalent width is obtained from numerical integration of the emergent line profile:

$$\tilde{W}_I = \int_{-\infty}^{+\infty} R_I(v, \eta_0, \alpha) \, \mathrm{d}\, v \,, \tag{75}$$

and $\tilde{W}_F = 5/6 \, \tilde{W}_I$. An 'analytical' curve-of-growth, $\tilde{W}(\eta_0; \alpha = 0.01)$ is shown in Fig. 1.

The equivalent width in [mA] is obtained from the reduced equivalent width by

$$W_\lambda[\mathrm{m\AA}] = 1000 \, \lambda_0[\mathrm{\AA}] \, \frac{\Delta v_D}{c} \, \tilde{W} \,. \tag{76}$$

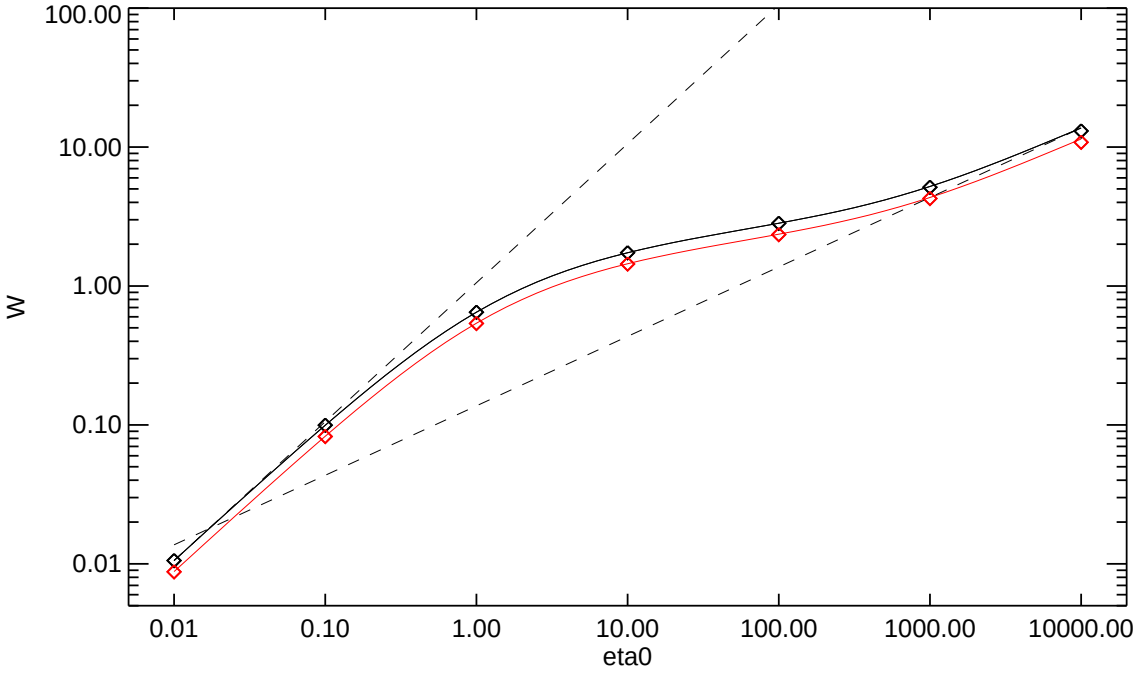

Figure 1: Analytical curve-of-growth showing the (reduced!) equivalent width (integrated from $v = -100$ to $v = +100$) as a function of η_0, assuming $\alpha = 0.01$. **Black**: disk-center, **red**: flux. The dashed lines have slopes 0.5 and 1.0. Diamonds show the numerical results obtained with LINFOR3D (integration from $v = -50$ to $v = +50$).

The results of a number of test calculations are listed below. The wavelength resolution was chosen to be 1/10 of the Doppler width: $\delta\lambda = 0.1 \, \lambda_0 \, \Delta v_D/c$. The wavelength range was set to

±50 Doppler widths; $\Delta v_D = 6$ km/s, $\alpha = 0.01$. The line file used for the test calculations is shown below.

```
alam      Vdop      eta0     avgt     dlam     ddlam
7    7
Test     grey sf   Vdop=2.D-5, eta0=1.0D-2, avgt=1.D-2
1    7
4000.000 2.0D-5    1.0D-2    1.0D-2   4.00D0   0.80D-2
Test     grey sf   Vdop=2.D-5, eta0=1.0D-1, avgt=1.D-2
1    7
4000.000 2.0D-5    1.0D-1    1.0D-2   4.00D0   0.80D-2
Test     grey sf   Vdop=2.D-5, eta0=1.0D0, avgt=1.D-2
1    7
4000.000 2.0D-5    1.0D0     1.0D-2   4.00D0   0.80D-2
Test     grey sf   Vdop=2.D-5, eta0=1.0D1, avgt=1.D-2
1    7
4000.000 2.0D-5    1.0D1     1.0D-2   4.00D0   0.80D-2
Test     grey sf   Vdop=2.D-5, eta0=1.0D2, avgt=1.D-2
1    7
4000.000 2.0D-5    1.0D2     1.0D-2   4.00D0   0.80D-2
Test     grey sf   Vdop=2.D-5, eta0=1.0D3, avgt=1.D-2
1    7
4000.000 2.0D-5    1.0D3     1.0D-2   4.00D0   0.80D-2
Test     grey sf   Vdop=2.D-5, eta0=1.0D4, avgt=1.D-2
1    7
4000.000 2.0D-5    1.0D4    1.0D-2   4.00D0   0.80D-2
clam         gfscale
-4000.000    1.0
```

For the following tabulations we have defined

$$\Delta W_I = \log_{10} W_I(\text{linfor3D}) - \log_{10} W_I(\text{Eq.(75)}\,, \tag{77}$$

and

$$\Delta W_F = \log_{10} W_F(\text{linfor3D}) - \log_{10} \frac{5}{6} W_I(\text{Eq.(75)}\,, \tag{78}$$

These results are obtained with `intline=1`:

η_0	ΔW_I	ΔW_F		
	[dex]	`ntheta=`-3	`ntheta=3`	`ntheta=4`
1.0E-02	+0.000342	+0.000336	−0.002949	−0.001690
1.0E-01	+0.000331	+0.000327	−0.002958	−0.001698
1.0E+00	+0.000269	+0.000271	−0.003014	−0.001755
1.0E+01	+0.000072	+0.000081	−0.003203	−0.001942
1.0E+02	−0.000820	−0.000807	−0.004088	−0.002831
1.0E+03	−0.005441	−0.005432	−0.008714	−0.007456
1.0E+04	−0.021428	−0.021421	−0.024704	−0.023446

These results are obtained with `intline=-2`:

η_0	ΔW_I	ΔW_F		
	[dex]	`ntheta=`-3	`ntheta=`3	`ntheta=`4
1.0E-02	+0.000334	+0.000328	−0.002957	−0.001698
1.0E-01	+0.000324	+0.000320	−0.002965	−0.001706
1.0E+00	+0.000261	+0.000263	−0.003022	−0.001762
1.0E+01	−0.000063	+0.000074	−0.003211	−0.001950
1.0E+02	−0.000825	−0.000814	−0.004097	−0.002838
1.0E+03	−0.005447	−0.005438	−0.008720	−0.007463
1.0E+04	−0.021432	−0.021425	−0.024708	−0.023450

4 Program Files and Data input files

In this section all the program files making up the LINFOR3D package are listed. First an overview on the program flow and the structures in common block `linfordata` is given. The format of the data input files is described since the primary way the user controls the program execution is via the control parameters read from `linfor_setcmd.pro` (see Section 5). The line parameters are specified in the input file `line.dat` (see Section 6).

4.1 Main program flow

Basically, the calling sequence is as follows (incomplete listing of linfor_3D.pro):

- Read input parameters (`linfor_setcmd.pro`)
- Initialize atomic data (`linfor_atom.pro`
- Read line data: (`linfor_rdline.pro`)
- Initialize ionopa abundances, opacity tables and EOS tables
- Set constants (`linfor_init`)
- Define `ff`, type linfor_flowfield (`linfor_flowfield__define.pro`)
- Define `f1`, type linfor_flowfield: (`linfor_flowfield__define.pro`)
- Define `fx`, type linfor_flowfield: (`linfor_flowfield__define.pro`)
- Define `ss`, type linfor_spectrum: (`linfor_spectrum__define.pro`)
- Define `s1`, type linfor_spectrum: (`linfor_spectrum__define.pro`)
- Define `sx`, type linfor_spectrum: (`linfor_spectrum__define.pro`)
- Read model data into ff structure (`linfor_rduio.pro`)
- Recompute model on refined z-grid (`linfor_regrid.pro`)
- Compute ionopa quantities (pe, kappa, zeta) and monochromatic tau for 3D model (`linfor_ionopa_3d.pro`)
- Construct 1D reference atmosphere from `ff`, store in `f1`: (`linfor_refatm.pro`)
- Compute ionopa quantities (pe, kappa, zeta) and monochromatic tau for 1D reference atmosphere (`linfor_ionopa_3d`)
- Do radiative transfer calculations for 3D model (`linfor_dort.pro`)
- Do radiative transfer calculations for averaged 3D atmosphere (`linfor_dort.pro`)
- Store results for later evaluation (`linfor_eval, ss, s1, nf, kl`)
- Make Plots of line profiles and bisectors (`linfor_plot1.pro`)
- Do radiative transfer calculations for 1D reference atmosphere (`linfor_dort.pro`)
- Store results for later evaluation (`linfor_evalx.pro`)
- Create postcript file(s) (`linfor_plot2.pro`)
- Generate output files `linfor_3D_1.idlsave`, `linfor_3D_2.idlsave`
- Free pointers to structures `ff`, `f1`, `fx`, `ss`, `s1`, and `sx` if `free_flag` = 1 (see Sect. 5.1) (`linfor_flowfield_free.pro`)

4.2 Structures in Common Block linfordata

Table 1 shows a list of the structrues in common block 'linfordata' used by the linfor_3D package.

Structure	Defined in	Description
`atom`	`linfor_atom.pro`	Atomic weights & ionization potentials
`const`	`linfor_init.pro`	Physical & model constants
`cmd`	`linfor_setcmd.pro`	Input parameters controlling program execution
`line`	`linfor_rdline.pro`	Line data derived from 'line.dat'
`gas`	`linfor_init.pro`	GAS tables initialized by 'tabinter_rdcoeff'
`eos`	`linfor_init.pro`	EOS tables initialized by 'tabinter_rdcoeff'
`result`	`linfor_init.pro`	Basic results for computing abundance corrections

Table 1: List of all structures in common block 'linfordata': the table shows the name of the structure, the routine where it is defined, and a description.

4.3 IDL Files

Table 3 shows a list of all source files necessary to run LINFOR3D. Finally, you will also need the files which are listed in Tab. 2:

File name	Type	Description
`blam.pro`	F	Computes the Kirchhoff-Planck function.
`monocubic.pro`	F	Performs monotonic piecewise cubic interpolation.
`ms_int.pro`	F	Integrates a given function over optical depth.

Table 2: List of additional IDL modules which not unique to LINFOR3D.

File name	Type	Description
`linfor_3D.pro`	S	main program
`linfor_flowfield__define.pro`	S	Definition of flow field structure
`linfor_spectrum__define.pro`	S	Definition of spectrum structure
`linfor_raysys__define.pro`	S	Definition of ray system structure
`linfor_atom.pro`	S	Defines atomic data
`linfor_setwts.pro`	S	Defines weights for angle quadrature
`linfor_setcmd.pro`	S	Command file, parameter input
`linfor_rdxatm.pro`	S	Reads 1D reference atmosphere, calling `linfor_rdatmos` or `linfor_rdatlas9`
`linfor_rdatlas9.pro`	S	Reads ATLAS9 1D atmosphere (atm.dat)
`linfor_rdatmos.pro`	S	Reads ATMOS 1D atmosphere (atm.dat)
`linfor_rdf15.pro`	S	Reads a sequence of FOR15 snapshots from 2D Kiel hydro simulations (FOR15)
`linfor_rdsav.pro`	S	Reads 3D snapshot from Copenhagen code (savfs)
`linfor_rduio.pro`	S	Reads 3D snapshot from CO^{5}BOLD uio output files
`linfor_rdvog.pro`	S	Reads 3D snapshot from Voegler MHD code
`linfor_findff.pro`	S	Finds cached flow fields
`linfor_rdline.pro`	S	Reads line data (line.dat)
`linfor_init.pro`	S	Initializes ionopa, EOS, Opacities, several constants
`linfor_bisector.pro`	S	Computes line bisector positions called by `linfor_plot1` and `linfor_plot2`
`linfor_convol.pro`	S	Convolves line profile with Gauss kernel called by `linfor_plot1` and `linfor_plot2`
`linfor_dort.pro`	S	Computes spectrum from flow field (main RT module calling several lower level routines)
`linfor_eval.pro`	S	Evaluates mean spectrum, "abundance corrections"
`linfor_evalx.pro`	S	Evaluates reference spectrum, "abundance corrections"
`linfor_incline.pro`	S	Inclines 3D flow field, called by `linfor_ztau`
`linfor_ionopa_3d.pro`	S	Calculates electron pressure, ionization fractions, and monochromatic optical depth for given flow field
`linfor_rad3.pro`	S	Integration of RT equation
`linfor_refatm.pro`	S	Define 1D reference atmosphere from 3D flow field
`linfor_regrid.pro`	S	Cut out surface layers from original model, re-define grid
`linfor_tauinfo.pro`	S	Prints information about optical depth scales
`linfor_ztau.pro`	S	Prepares bundle of (inclined) rays on monochromatic tau
`linfor_plot0.pro`	S	Plots flow field
`linfor_plot1.pro`	S	Plots spatially resolved line profiles
`linfor_plot2.pro`	S	Plots averaged line profiles
(`linfor_plot3.pro`)	S	Plots monochromatic granulation images
`alpha_line.pro`	F	Computes α-parameter for VOIGT function
`eta0.pro`	F	Computes η_0, the opacity at line center of metal lines
`rrca.pro`	F	Computes mean square orbital radius of electron (Unsöld)
`vdop.pro`	F	Computes (thermal+turbulent) Doppler velocity [c_s]
`linfor_timing.pro`	S	Prepares and gathers timing statistics.
`linfor_timing_print.pro`	S	Print timing statistics

Table 3: List of all IDL modules: the table shows the file name, the type (**S**ubroutine or **F**unction, and its description.

5 Parameter Input: linfor_setcmd.pro

The input parameters (except for those defined in `line.dat`, see Sect. 6) are basically specified by editing the routine `linfor_setcmd.pro`. In this way, the user defines the structure `cmd` (see Table 1). The order of entries is irrelevant. Parameters which are not required may be omitted. A detailed explanation of the various input parameters and their possible values is given in the following sections. An example follows in Sect. 5.9.

5.1 Program execution flags

The user can control the program execution by setting the flags `run_flag`, `nlte_flag`, `cv1_flag`, `cv2_flag`, `cv3_flag`, `plt_flag`, `maps_flag`, `cc3d_flag`, `rdbb_flag`, `free_flag`, which are explained in more detail below.

5.1.1 run_flag

function	:	program mode
required	:	always
type	:	integer
values	:	−2, −1, 0, 1, 2, 3 (usually 3)

This parameter determines the general function of LINFOR3D:

Setting `run_flag =-2` allows you to compute 3x3 file for the external reference model only.

Setting `run_flag =-1` allows you to restore old results, and replace the results of the previous 1D external atmosphere with those of a different 1D external atmosphere.

Setting `run_flag = 0` (similar to run_flag = -1) allows you to quickly compare the 3D spectra with another external 1D reference atmosphere. Finally, the results are saved in files '`linfor_3D_1.idlsave`' and '`linfor_3D_2.idlsave`'. Rarely used setting.

Setting `run_flag = 1` is used for plotting the structure of the input model on the original grid. No radiative transfer calculations are done.

Setting `run_flag = 2` is used for plotting the structure of the input model on the reduced (refined) grid. No radiative transfer calculations are done.

Setting `run_flag = 3` is the usual case. After construction of the 3D atmosphere on the reduced (refined) grid and of the 1D mean atmosphere, the line formation calculations are done, and the results are plotted ('linfor_plot1': spatially and temporally resolved line profiles and bisectors, 'linfor_plot2': surface and time averaged line profiles and bisectors). Finally, the results are saved in files '`linfor_3D_1.idlsave`' and '`linfor_3D_2.idlsave`'.

run_flag value		control of program flow
-1	:	restore results, (compute 1D ref. atmosphere & spectrum), save results
0	:	restore results, (compute 1D ref. atmosphere & spectrum), plot2, save results
1	:	compute 3D, 1D atmospheres (1), plot01, stop
2	:	compute 3D, 1D atmospheres (1,2), plot02, stop
3	:	compute 3D, 1D atmospheres (1,2), line formation, plot1, plot2, save results

5.1.2 cv1_flag

function	:	enforce $\langle \rho\, u_x \rangle = 0$
required	:	always
type	:	integer
values	:	0, 1

The parameter cv1_flag controls whether or not the x-component of the velocity field is adjusted to ensure zero mass flux in x-direction. (0: no, 1: yes). Default 0

5.1.3 cv2_flag

function	:	enforce $\langle \rho\, u_y \rangle = 0$
required	:	always
type	:	integer
values	:	0, 1

The parameter cv2_flag controls whether or not the y-component of the velocity field is adjusted to ensure zero mass flux in y-direction. (0: no, 1: yes). Default 0

5.1.4 cv3_flag

function	:	enforce $\langle \rho\, u_z \rangle = 0$
required	:	always
type	:	integer
values	:	0, 1

The parameter cv3_flag controls whether or not the z-component of the velocity field is adjusted to ensure zero mass flux in z-direction. (0: no, 1: yes). Default 0

5.1.5 plt_flag

function	:	plotting of bisectors
required	:	always
type	:	integer
values	:	-1, 0, 1

The parameter plt_flag controls if line bisctors should be plotted or not (0: no, 1: yes). If plt_flag is set to -1, all plotting is suppressed.

5.1.6 maps_flag

function	:	controls output of intensity maps
required	:	always
type	:	integer
values	:	0, 1, 2

The parameter `maps_flag` controls the output of intensity maps which are provided in the IDL structure `MAPS`:

value		meaning
0	:	Continuum images only. Create map `ICLAM0`.
1	:	Continuum images (`ICLAM0`) plus images at the centre of the wavelength window (`ICLAM1`), all at wavelength $\lambda =$`clam`;
2	:	Continuum images (`ICLAM0`) plus images (`ICLAM2`) at all wavelengths within the wavelength window of width $2 \cdot$ `dlam` around the central wavelength `clam`: $\lambda_i =$ `clam` $-$ `dlam` $+ i \cdot$ `ddlam` (see Sect. 6);

5.1.7 cc3d_flag

function	:	output of 3D contribution function
required	:	always
type	:	integer
values	:	0, 1

The parameter `cc3d_flag` controls whether the 3D continuum intensity contribution function should be saved in structure contf3d or not (0: no, 1: yes).

5.1.8 nlte_flag

function	:	output of 3D contribution function
required	:	always
type	:	integer
values	:	0, 1, 2, 3

The parameter `nlte_flag` controls whether the line transfer is performed in LTE (`nlte_flag`=0) or in NLTE (`nlte_flag`=1, 2, 3). The NLTE options work only for lines with available departure coefficients, which are read from a separate data file (see below).

value		meaning
0	:	Continuum and lines in LTE.
1	:	Continuum in LTE, line source function in LTE, line opacity in NLTE
2	:	Continuum in LTE, line opacity in LTE, line source function in NLTE,
3	:	Continuum in LTE, line opacity and source function in NLTE

5.1.9 free_flag

function	:	free pointers in structures at end of program
required	:	always
type	:	integer
values	:	0, 1

If `free_flag = 1`, then each run of LINFOR3D allocates fresh memory for the structures `ff`, `f1`, `fx`, `ss`, `s1`, and `sx`. In this case the corresponding pointers are removed at the end. If you want to examine the structures after the end of execution, you must have `free_flag = 0`. If you want to run the program several times in a row with different input parameters, you should set `free_flag = 0` in order to avoid additional memory allocation for each run.

5.2 General paths

5.2.1 abupath

function	:	directory where '`.abu`' files and '`atom.dat`' are located
required	:	always
type	:	string
values	:	e.g. '`/home/mst/ABU/`'

If `abupath` is not specified in the command file, the path is taken from environment variable '`LINFOR3D_ABU`'.

5.2.2 ff_path

function	:	directory to be used for reading and writing cached flow fields
required	:	always
type	:	string
values	:	e.g. '`/data/mst/ffcache/`'

5.2.3 opapath

function	:	directory with opacity tables (.opta files)
required	:	always
type	:	string
values	:	e.g. '`/home/mst/RHD/opa/dat/`'

5.2.4 gaspath

function	:	directory with GAS tables (gas_*.eos files)
required	:	always
type	:	string
values	:	e.g. '`/home/mst/RHD/eos/dat/`'

5.2.5 eospath

function	:	directory with EOS tables (eos_*.eos files)
required	:	always
type	:	string
values	:	e.g. '`/home/mst/RHD/eos/dat/`'

5.3 Model data

5.3.1 context

function	:	source of input model
required	:	always
type	:	string
values	:	e.g. '`cobold`'

value		meaning
'`cobold`'	:	3D CO^{5}BOLD
'`copenhagen`'	:	N&S 3D code
'`kiel`'	:	Kiel 2D HDW-Code
'`muram`'	:	MURAM 3D MHD Code
'`grey`'	:	construct grey 3D ($n_x = n_y = 10$)
	:	hydrostatic atmosphere for test purposes

The $\tau_{\rm Ross}$ grid of the grey atmosphere is defined by the parameters `cmd.lutau1`, `cmd.lutau2`, `cmd.dlutau`. The atmospheric parameters must be specified as `cmd.Teff` and `cmd.grav`. The opacity table must be specified as `cmd.opafile`, and the equation of state as `cmd.eosfile` and `cmd.gasfile`.

5.3.2 rhdpath

function	:	directory with 2D/3D model atmospheres (.end, .full files)
required	:	always
type	:	string
values	:	e.g. '`/data/mst/model/`'

5.3.3 modelid

function	:	name of 2D/3D model file
required	:	always
type	:	string
values	:	e.g. '`gt57g44n66_3Dgz.end`'

Note: A list of files can be specified by using wildcards, e.g. '`chro3D04*.full`'.

5.3.4 parfs

function	:	full path to parameter file (rhd.par)
required	:	CO5BOLD only
type	:	string
values	:	e.g. '`/data/mst/model/par/gt57g44n66.par`'

5.3.5 xbcpath

function	:	full path to xbc files (*.xbc)
required	:	CO5BOLD only
type	:	string
values	:	e.g. '`/data/mst/NLTE3D_data/model/`'

Note: xbc-files are necessary for NLTE line formation calculations. Presently limited to Li for selected CO5BOLD models.

5.3.6 abuid

function	:	Abundance mixture to be used in `ionopa`-routines
required	:	always
type	:	string
values	:	'kiel', 'cifist2006', 'special'

`abuid` identifies the solar abundance mix which is then modified according to `dmetal` and `dalpha` (see below). The corresponding tables, `kiel.abu`, `cifist2006.abu`, or `special.abu` must be located in diectory `abupath`.

5.3.7 dmetal

function	:	metallicity [M/H] ($\log_{10}$) to be used in `ionopa`-routines
required	:	always
type	:	float
values	:	e.g. 0.0, -0.5, -2.0

The logarithmic abundance of all elements beyond Li ($N > 3$) is changed by `dmetal`.

5.3.8 dalpha

function	:	alpha enhancement to be used in `ionopa`-routines
required	:	always
type	:	float
values	:	e.g. 0.0, +0.4

The logarithmic enhancement factor to be applied to all α-elements (O Ne Mg Si S Ar Ca Ti).

5.3.9 nx_skip

function	:	sampling of model in x-direction
required	:	if `context`='`cobold`','`kiel`','`muram`'
type	:	integer
values	:	1, 4, 10; -1

If both `nx_skip` and `ny_skip` (see Sect. 5.3) are negative, the original data are re-binned from (nx,ny) to `(nx/abs(nx_skip),ny/abs(ny_skip))`. In the usual case that both `nx_skip` and `ny_skip` are positive, the original data are re-sampled, skipping by `nx_skip` in x, and by `ny_skip` in y-direction (`nx/nx_skip`, and `ny/ny_skip` should preferably be an integer). If `nx_skip` and `ny_skip` have different signs, an error message is printed and the program is stopped. The value 1 has no effect.

5.3.10 ny_skip

function	:	sampling of model in x-direction
required	:	if `context`='`cobold`','`kiel`','`muram`'
type	:	integer
values	:	1, 4, 10; -1

For details see description of `nx_skip` (Sect. 5.3).

5.4 More model information (all read from parameter file for CO5BOLD data)

The parameters in this section are ignored in the case of CO5BOLD data and instead read from the specified CO5BOLD parameter file.

5.4.1 opafile

function	:	name of opacity file (binned opacity tables)
required	:	not needed if `context`='`cobold`'
type	:	string
values	:	e.g. '`g2v.opta`'

5.4.2 gasfile

function	:	name of GAS file ($P, T \rightarrow \rho, e, \ldots$)
required	:	not needed if `context='cobold'`
type	:	string
values	:	e.g. '`gas_mm00_l.eos`'

5.4.3 eosfile

function	:	name of EOS file ($\rho, e \rightarrow P, T, \ldots$)
required	:	not needed if `context='cobold'`
type	:	string
values	:	e.g. '`eos_mm00_l.eos`'

5.4.4 htau0

function	:	opacity scale height [cm] at top of 3D model
required	:	not needed if `context='cobold'`
type	:	float
values	:	e.g. $60.0E5$

5.4.5 qmol

function	:	mean molecular weight of neutral gas
required	:	not needed if `context='cobold'`
type	:	float
values	:	e.g. 1.301855

5.4.6 Teff

function	:	effective temperature of 3D model
required	:	not needed if `context='cobold'`
type	:	float
values	:	e.g. 5770.0

5.4.7 grav

function	:	surface gravity [cm/s^2] of 3D model
required	:	not needed if `context='cobold'`
type	:	float
values	:	e.g. 27500.0

5.4.8 tsurffac

function	:	surface temperature ($\tau = 0$) of 3D model is `tsurffac`$\cdot T_{\rm eff}$
required	:	not needed if `context='cobold'`
type	:	float
values	:	e.g. 0.727903

5.5 Model data - reading of 'full' files (CO5BOLD only)

The parameters in this section are only needed for reading snapshot from CO5BOLD data files.

5.5.1 isnap_full_1

function	:	first snapshot to be read from full file(s)
required	:	only needed if `context='cobold'`
type	:	integer
values	:	1

5.5.2 isnap_full_2

function	:	last snapshot to be read from full file(s)
required	:	only needed if `context='cobold'`
type	:	integer
values	:	1

5.5.3 istep_full

function	:	step for reading snapshots from full file(s)
required	:	only needed if `context='cobold'`
type	:	integer
values	:	1

5.6 $\langle 3D \rangle$ mean model

5.6.1 mavg

function	:	mode of averaging 3D T-structure on τ_{Ross}
required	:	always
type	:	integer
values	:	1, 4

value		meaning
1	:	$T_{\langle 3D \rangle}(\tau_{\mathrm{Ross}}) = \langle T_{\mathrm{3D}}(\tau_{\mathrm{Ross}}) \rangle$
4	:	$T_{\langle 3D \rangle}(\tau_{\mathrm{Ross}}) = \langle T^4_{\mathrm{3D}}(\tau_{\mathrm{Ross}}) \rangle^{1/4}$

5.7 External 1D reference model

5.7.1 atmpath

function	:	directory with 1D model atmospheres
required	:	always
type	:	string
values	:	e.g. '`/home/mst/atm/`'

5.7.2 atmfile

function	:	name of 1D reference model
required	:	always
type	:	string
values	:	e.g. '`NONE`', '`falc.atm`'

Note: No external 1D reference atmosphere will be used if `atmfile`='`NONE`'. In this case the parameter `atmpath` has no meaning.

5.8 Line data and radiative transfer

5.8.1 linfs

function	:	name of line data file
required	:	always
type	:	string
values	:	e.g. '`Li67.line`'

Note: If `linfs` is not specified, the default value '`line.dat`' is assumed.

5.8.2 lutau1

function : smallest $\log \tau_{\rm Ross}$ covered by sub-model (refined z-grid)
required : always
type : float
values : e.g. $-7.0D0$

5.8.3 lutau2

function : largest $\log \tau_{\rm Ross}$ covered by sub-model (refined z-grid
required : always
type : float
values : e.g. $2.0D0$

5.8.4 dlutau

function : z-spacing of sub-model corresponds roughly to $\Delta \log \tau_{\rm Ross}$ =`dlutau`
required : always
type : float
values : e.g. $8.0D-2$

5.8.5 lctau1

function : smallest $\log \tau_{\rm cont}$ used for RT integration
required : always
type : float
values : e.g. $-7.0D0$, $\geq$ `lutau1`

5.8.6 lctau2

function : largest $\log \tau_{\rm cont}$ used for RT integration
required : always
type : float
values : e.g. $2.0D0$, $\leq$ `lutau2`

5.8.7 dlctau

function : resolution in $\log \tau_{\rm cont}$ used for RT integration
required : always
type : float
values : e.g. $8.0D-2$

5.8.8 ntheta

function : number of θ-angles for which spectrum is computed
required : always
type : integer
values : 0, 1, 2, 3, (-3), 4, 6, 8

0: Intensity spectrum, >0: Intensity and flux spectrum;

5.8.9 nphi

function	:	number of ϕ-angles for integration of flux spectrum
required	:	always
type	:	integer
values	:	no restriction, typically 4

5.8.10 mu0

function	:	view angle $\mu = \cos\theta$
required	:	always
type	:	float
values	:	0.0 .. 1.0

If the parameter `ntheta`=0, then the spectrum and intensity maps are computed for inclination angle `mu0` ($= \cos\theta_0$).
`mu0`= 1.0 corresponds to vertical rays, i.e. disk center view.
`mu0`= 0.0 corresponds to the very limb, but a value of mu0=0.0 will clearly not work.

5.8.11 kphi

function	:	view angle
required	:	always
type	:	integer
values	:	0, 1, 2, 3

The parameter `kphi` determines the direction from which the model is viewed:

value		meaning
0	:	rays emerge parallel to the x-axis, i.e. the model is viewed somewhere on the 'equator' between the left limb and disk center.
1	:	rays emerge parallel to the y-axis, i.e. the model is viewed somewhere on the 'meridian' between the lower limb and disk center.
2	:	rays emerge anti-parallel to the x-axis, i.e. the model is viewed somewhere on the 'equator' between the right limb and disk center.
3	:	rays emerge anti-parallel to the y-axis, i.e. the model is viewed somewhere on the 'meridian' between the upper limb and disk center.

Other (integer) values of kphi are allowed, but give no new results; increasing kphi by one increases phi by $\pi/2$.

5.8.12 Hbrd

function	:	controls broadening of hydrogen lines
required	:	always
type	:	integer
values	:	0, 1, 2, 3

value		meaning
0	:	Cayrel & Traving (1960), default
1	:	Resonance broadening: AG , Stark boradening: G
2	:	Resonance broadening: BPO, Stark boradening: G
3	:	Resonance broadening: A08 , Stark boradening: G

AG : Ali & Griem (1966, Phys. Rev. 144, 366),
BPO: Barklem, Piskunov and O'Mara (2000, A&A 363, 1091),
A08 : Allard et al. (2008, A&A 480, 581),
G : Griem (1960, ApJ 132, 883), with corrections to approximate the Vidal, Cooper & Smith (1973, ApJS 25, 37) profiles.

Note: option Hbrd=2 has an effect **only on Hα, Hβ, and Hγ**, and Hbrd=3 affects **only Hα**; all other hydrogen lines are treated according to option Hbrd=1, unless Hbrd=0.

5.8.13 ximicx

function	:	isotropic Gaussian microturbulence velocity [km/s] for external 1D reference model (added quadratically to thermal velocity)
required	:	always
type	:	float
values	:	e.g. 1.0

5.8.14 ximic1

function	:	isotropic Gaussian microturbulence velocity [km/s] for $\langle 3D \rangle$ mean models (added quadratically to thermal velocity)
required	:	always
type	:	float
values	:	e.g. 1.0

5.8.15 ximic3

function	:	isotropic Gaussian microturbulence velocity [km/s] for 2D/3D models (added quadratically to thermal flow velocity)
required	:	always
type	:	float
values	:	e.g. 1.0

5.8.16 ximacx

function	:	Isotropic Gaussian macroturbulence velocity [km/s] for external 1D reference model (additional line broadening after line formation)
required	:	always
type	:	float
values	:	e.g. 1.6

5.8.17 ximac1

function	:	Isoptropic Gaussian macroturbulence velocity $[km/s]$ for $\langle 3D \rangle$ mean models (additional line broadening after line formation)
required	:	always
type	:	float
values	:	e.g. 1.6

5.8.18 ximac3

function	:	Isotropic Gaussian macroturbulence velocity $[km/s]$ for 2D/3D models (additional line broadening after line formation)
required	:	always
type	:	float
values	:	e.g. 1.6

5.8.19 vfacx

function	:	the x-component of the hydrodynamical velocity field of the 2D/3D models is multiplied by this factor
required	:	always
type	:	float
values	:	e.g. $0.0, 1.0$

5.8.20 vfacy

function	:	the y-component of the hydrodynamical velocity field of the 2D/3D models is multiplied by this factor
required	:	always
type	:	float
values	:	e.g. $0.0, 1.0$

5.8.21 vfacz

function	:	the z-component of the hydrodynamical velocity field of the 2D/3D models is multiplied by this factor
required	:	always
type	:	float
values	:	e.g. $0.0, 1.0$

5.8.22 micro

function	:	controls microturbulence in 1D curve-of-growth
required	:	always
type	:	integer
values	:	0, 1

Determines whether or not different microturbulence values should be used when computing the 1D curve-of growth. 0: only one value, given by `ximicx` and `ximic1`, respectively; 1: sequence

of microturbulence values defined by parameters `xi_a`, `xi_b`, `xi_d` (see below).

5.8.23 xi_a

function	:	determines start value for microturbulence sequence
required	:	always
type	:	float
values	:	e.g. 0.0, default: 0.5

5.8.24 xi_b

function	:	determines end value for microturbulence sequence
required	:	always
type	:	float
values	:	e.g. 2.0, default: 1.5

5.8.25 xi_d

function	:	determines intervals of microturbulence sequence
required	:	always
type	:	float
values	:	e.g. 0.1, default: 0.125

The microturbulence sequence is computed as xi(i) = xi0 * (`xi_a` + i * `xi_d`), i=0 .. im, where xi0 is `ximicx` and `ximic1`, respectively, and im= (`xi_b` - `xi_a`)/ `xi_d`.

5.8.26 dclam

function	:	determines the variation of the continuum
required	:	always
type	:	float
values	:	e.g. 20.0, default: 0

if `dclam`=0, the continuum is treated as constant (default). Otherwise, the continuum is computed at 3 wavelength points, `clam-dclam`, `clam`, `clam+dclam`, where `clam` is the central wavelength (in Å) of the computed spectral range (see Sect. 6), and `dclam` is half the width of the specified spectral range (in Å). The continuum is computed by parabolic interpolation inside the spectral window. If the spectral range of the specified synthetic spectrum (which is defined by the parameters of the line file (see Sect. 6) exceeds a few Å, `dclam` should be set to match half the total spectral range.

5.8.27 intmode

function	:	mode of integration in routines `ms_int_tau` and `ms_int_exp`
required	:	always
type	:	integer
values	:	0, 1

Determines the mode of integration in routines `ms_int_tau` and `ms_int_exp`, which can be linear

(0) or monotonic and cubic (1, standard).

5.8.28 intline

function	:	mode of integrating the line transfer equation
required	:	always
type	:	integer
values	:	1, 2, −1, −2

Determines the method of integrating the line transfer equation (see Section 3 for details). Default value is `intline`=1.

value		meaning
1	:	Line depression on fixed $\log \tau$ scale (Eq. 41)
2	:	Line depression on monochromatic τ scale (Eq. 36)
-1	:	Line intensity on fixed $\log \tau$ scale (Eq. 25)
-2	:	Line intensity on monochromatic τ scale (Eq. 26)

5.8.29 nchunk

function	:	rad.transfer is done in `n_chunk` "slices"
required	:	always
type	:	integer
values	:	e.g. 2

Default is `nchunk = 1`, i.e. the whole model is processed as one block. For large models, it may be necessary to split the computation into several 'chunks' to save memory.

5.9 Example

```
pro linfor_setcmd
common linfordata

cmd = {$
;------------------------------------------------------------------------------
; Program execution flags:
nlte_flag: 0, $                ; 0 / 1 / 2 / 3: LTE or NLTE for lines with xb
 run_flag: 3, $                ; execution mode: -2, -1, 0, 1, 2, 3
 cv1_flag: 1, $                ; 0 / 1: enforce <rho*v1>(z)=0 off / on
 cv2_flag: 1, $                ; 0 / 1: enforce <rho*v2>(z)=0 off / on
 cv3_flag: 0, $                ; 0 / 1: enforce <rho*v3>(z)=0 off / on
 plt_flag: 1, $                ; -1 / 0 / 1: plotting off / bisectors off / on
maps_flag: 1, $                ; create maps ICLAM0 .. ICLAMm, m=map_flag
cc3d_flag: 0, $                ; 0 / 1: output of CC3(nx,ny,nx) off / on
rdbb_flag: 0, $                ; 0 / 1: Read magnetic field, write SIR output
free_flag: 0, $                ; free pointers in structures at end of program
;
;------------------------------------------------------------------------------
; General paths:
abupath: '/home/mst/idl/spesyn/Linfor_3D/ABU/', $
         ; Path to abu files and atom.dat
         ; if not set, abupath is read from environment variable LINFOR3D_ABU
ff_path: '/work2/mst/ffcache/', $   ; directory with cached flow fields
;                                     'NONE': do not use cached flow fields
opapath: '/home/mst/opta/', $       ; directory with opacity tables
gaspath: '/work2/mst/eos/dat/', $   ; directory with GAS tables
eospath: '/work2/mst/eos/dat/', $   ; directory with EOS tables
;
;------------------------------------------------------------------------------
; Model data:
context: 'cobold', $
rhdpath: '/work1/mst/dgt57g44n73/end/', $         ; directory with model data
modelid: 'gt57g44n73_3D[c-d]z.end', $                        ; data file name
parfs:   '/work1/mst/dgt57g44n73/par/gt57g44n73_3D_d.par', $ ; parameter file
xbcpath: '/work2/mst/NLTE3D_data/', $   ; directory of departure coefficients
abuid: 'cifist2006', $            ; abundance mixtures 'kiel', or 'cifist2006'
dmetal: 0.0, $                    ; log10 scaling for metal abundances (Z>3)
dalpha: 0.0, $                    ; log10 scaling for alpha elements
;                                   (O, Ne, Mg, Si, S, Ar, Ca, Ti)
nx_skip: 5, ny_skip: 5, $   ; sampling in x, y (kiel, cobold only)
;
; more information (all read from parameter file for CO5BOLD)
opafile: 'undefined', $
gasfile: 'undefined', $
eosfile: 'undefined', $
htau0:  00.0E5, $
qmol:   00.0E5, $                   ; mean molecular weight
teff: 5770.0, grav: 27500.0, $     ; copenhagen only
tsurffac: 0.727903, $               ; Surface temperature = tsurffac*Teff
;------------------------------------------------------------------------------
; Reading of 'full' files (CO5BOLD only):
isnap_full_1: 1, $             ;  first snapshot to be read from full file(s)
isnap_full_2: 9, $;            ;   last snapshot to be read from full file(s)
istep_full:   2  $             ; step for reading snapshots from full file(s)
;------------------------------------------------------------------------------
; <3D> mean model:
mavg: 4, $      ; 1: T-average, 4: T^4-average for defining <3D> atmosphere
```

```
;-------------------------------------------------------------------------
; External 1D reference model:
atmpath: '/home/mst/idl/spesyn/1Dmodels/HM/', $ ; directory of reference model
atmfile:  'atm50_shm10c.atm', $                 ; name of reference model
;                                                 'NONE': no reference model
;-------------------------------------------------------------------------
; Line data / radiative transfer:
linfs:   'line.dat', $            ; File with line data
lutau1: -7.0D0, lutau2: 2.0D0, dlutau: 8.0D-2, $ ; tau scale defining vertical
;                                                  model extent and resolution
lctau1: -5.0D0, lctau2: 2.0D0, dlctau: 8.0D-2, $ ; universal tau scale for
;                                                  integration of RT equation
ntheta: 0, nphi: 4, $           ; number of theta and phi angles
mu0: 0.40, kphi: 0, $           ; view angle if ntheta=0 (cos theta,kphi*pi/2)
n_chunk:  1,  $                 ; RT is done in n_chunk "slices"
;-------------------------------------------------------------------------
Hbrd:    '0', $                  ; option for H line broadening
                                 ; 0 - (default) Cayrel & Traving
                                 ; 1 - BPO (self reson) + Griem (stark)
                                 ; 2 - Ali-Griem (self reson) + Griem (stark)
ximicx: 1.00, $                  ; microturbulence [km/s], 1D-REF atmosphere
ximic1: 1.00, $                  ; microturbulence [km/s], 1D-AVG atmosphere
ximic3: 0.00, $                  ; microturbulence [km/s], 3D-RHD atmosphere
ximacx: 1.60, $                  ; macroturbulence [km/s], 1D-REF atmosphere
ximac1: 1.60, $                  ; macroturbulence [km/s], 1D-AVG atmosphere
ximac3: 0.00, $                  ; macroturbulence [km/s], 3D-RHD atmosphere
vfacx:  1.00, $                  ; fudge factor for 3D x-velocity
vfacy:  1.00, $                  ; fudge factor for 3D y-velocity
vfacz:  1.00, $                  ; fudge factor for 3D z-velocity
micro:    0, $                   ; compute microturbulence sequence (0/1)
xi_a: 0.0, xi_b: 2.0, xi_d: 0.1, $  ; start, stop, delta of micro sequence
dclam:  0.0, $  ; variation of continuum from clam-dclam .. clam+dclam [A]
intmode:  1, $                     ; integration mode (linfor_msint)
intline:  1, $                     ; line integration: depth (1,2) / I (-1,-2)
;
}
;
end
```

6 Line Data File: line.dat

There are several different formats (for historical reasons) to specify line data which are described in Sect. 6.2.
Note that all formats were extended in version 1.5.0 and now do have to contain the two lines

```
clam     gfscale
2000.0   1.0
```

at the end. These parameters are explained in Sect. 6.1. Some helpful remarks concerning the conversion of line broadening parameters are given in Sect. 6.3.

6.1 Parameters in Line Data File

6.1.1 clam

function	:	continuum wavelength in Å, also center of wavelength window
required	:	always
type	:	float
values	:	e.g., 2000.0

`clam` defines the wavelength where the continuum opacities are computed, and also defines the center of the window for which spectrum synthesis is done. The window extends from $\lambda =$`clam-dlam` to $\lambda =$`clam+dlam`, depending on the value of `dlam` specified for the particular line.
From Version 3.1.2, a **negative** `clam` indicates that the continuum source function is to be set to the wavelenth-integrated Planck-Function, $S = \sigma T^4/\pi$, and the continuum opacity is set to the Rosseland mean opacity, $\kappa_{\rm cont} = \kappa_{\rm Ross}$..

6.1.2 gfscale

function	:	global scaling factor for oscillator strengths
required	:	always
type	:	float
values	:	e.g., 1.0

Note: The value 1 has no effect.

6.2 Line Data Formats

6.2.1 Continuum only

It is possible to do pure continuum calculations. In this case, the 'line.dat' file looks like this.

Example:

```
Some text header
1   1
Continuum, 2000 A
1   -1
clam     gfscale
2000.0   1.0
```

Description of entries:

Row 1: Header (identifies the meaning of the columns for data in row 5)
Row 2: Two integers, *kline* and *ktotal*; both of them must be 1

Row 3: String, identifier of the continuum calculation
Row 4: Two integers, $nbl = 1$, $incode = -1$
Row 5: Description for data in row 6
Row 6: `clam` and `gfscale` (see Sect.6.1)

All the *line* parameters remain undefined.

6.2.2 Single line calculations, line data format '0'

For a **single unblended line**, the simplest form of the 'line.dat' file looks like this.

Example:

```
Mult   namj      ei      alam     gflg     dlgC6   drrca1  dlam   ddlam
1   1
Fe I, 5500 A, 0.00 eV
1   0
0000   2600      0.000   5500.0   -6.000   1.0     10.0    5.5D-1 5.5D-3
clam     gfscale
2000.0   1.0
```

Description of entries:

Row 1: Header (identifies the meaning of the columns for data in row 5)
Row 2: Two integers, *kline* and *ktotal*
kline: number of line calculations requested in this file
ktotal: is the total number of spectral lines including blends
in this case $kline = 1$, $ktotal = 1$
Row 3: String, identifier of the (first) line calculation
Row 4: Integer *nbl*, integer array *incode*(*nbl*)
nb: number of blend components for this line calculation (= 1)
incode: integer array identifying the input format for each of the blend components (= 0)
Row 5: Line data in format '0' (7 + 2 columns):
C1: Multiplet number (for information only)
C2: Identifier of atom or ion (e.g. 2601 mean FeII)
C3: Excitation potential of lower level in [eV]
C4: Central wavelength of blend component
C5: $\log gf$ value of blend component
C6: $\Delta \log C_6$: Enhancement factor for van der Waals line broadening
C7: $\Delta\overline{r^2}/a_0^2$: Difference of mean square electron orbital radii
C8: $\Delta\lambda$ [Å]: Line profile is computed from $\lambda_0 - \Delta\lambda$ to $\lambda_0 + \Delta\lambda$
C9: $\delta\lambda$ [Å]: Spacing of wavelength points for spectrum synthesis
(C10: W_0 [mÅ]: total equivalent width of this blend, see below)
Row 6: Description for data in row 6
Row 7: `clam` and `gfscale` (see Sect.6.1)

In this case, the Stark broadening (due to collisions with electrons) is neglected ($C_4 = 0$). Radiative damping ($\gamma_{\rm rad}$) is treated in the classical approximation.

In the case of a **single blended line** the 'line.dat' file looks as follows:

Example:

```
Mult   namj      ei      alam     gflg     dlgC6   drrca1  dlam   ddlam
1   2
Fe I, 0.00 eV + Fe II, 3.00 eV, 2000 A
2   0 0
9999   2600      0.000   2000.0   -6.441   1.0     10.0
9999   2601      3.000   2000.0   -4.550   1.0     10.0    1.5D-1 1.5D-3
clam     gfscale
2000.0   1.0
```

Note that it is not necessary that the blend components belong to the same ion. Here $kline = 1$, $ktotal = 2$, $nbl = 2$, $incode = [0, 0]$. Note that only the last of the rows describing the blend need entries $C8$ and $C9$.

With a slight modification, it is possible to enter an **equivalent width** (W_0 in [mÅ]) in column $C10$. For this purpose, nbl must be negative, with $|nbl|$ being the number of blend components. The gf value producing this equivalent width W_0 is returned in `result.gflg01` (average 3D atmosphere) and `result.gflg0x` (1D reference atmosphere).

Example unblended line:

```
Mult   namj  chik    alam      gflg    dlgC6  drrca1  dlam     ddlam      W0
  1  1
  N I Fictitious Line  1: /   0.000   5500.0  -7.6914  1.00   10.00   75.00 /
 -1  0
9999   700   0.000   5500.0  -7.6914  1.00   10.00  3.00E-01  3.00E-03   75.00
clam     gfscale
2000.0   1.0
```

Example blended line:

```
Mult   namj  chik    alam      gflg    dlgC6  drrca1  dlam     ddlam      W0
  1  2
Fe I, 0.00 eV + Fe II, 3.00 eV, 2000 A
 -2  0 0
9999   2600  0.000   2000.0   -6.441   1.0    10.0
9999   2601  3.000   2000.0   -4.550   1.0    10.0  1.50D-1   1.50D-03  100.00
clam     gfscale
2000.0   1.0
```

6.2.3 Single line calculations, line data format '1'

For a **single unblended line**, the this form of the 'line.dat' file looks like this.

Example:

```
Mult   namj   ei      alam     gflg     dlgC6 lu  diu  lo  dio  dlam   ddlam
1   1
O I ApJ Line 2: 92  6300.30  0.000 -9.773
1   1
  92   800     0.000  6300.30  -9.773   1.0   1   0.0  2   0.0  4.D-1  4.D-3
clam     gfscale
2000.0   1.0
```

Description of entries:

Row 1: Header (identifies the meaning of the columns for data in row 5)
Row 2: Two integers, *kline* and *ktotal*

kline: number of line calculations requested in this file
ktotal: is the total number of spectral lines including blends
in this case $kline = 1$, $ktotal = 1$
Row 3: String, identifier of the (first) line calculation
Row 4: Integer *nbl*, integer array *incode*(*nbl*)
nb: number of blend components for this line calculation (= 1)
incode: integer array identifying the input format for each of the blend components (= 1)
Row 5: Line data in format '1' (10 + 2 columns):
C1: Multiplet number (for information only)
C2: Identifier of atom or ion (e.g. 2601 mean FeII)
C3: Excitation potential of lower level in [eV]
C4: Central wavelength of blend component
C5: $\log gf$ value of blend component
C6: $\Delta \log C_6$: Enhancement factor for van der Waals line broadening
C7: *LU*: Orbital quantum number of valence electron of lower level
C8: *DIU*: excitation energy [*eV*] of parent term for lower level
C9: *LO*: Orbital quantum number of valence electron of upper level
C10: *DIO*: excitation energy [*eV*] of parent term for upper level
C11: $\Delta\lambda$ [Å]: Line profile is computed from $\lambda_0 - \Delta\lambda$ to $\lambda_0 + \Delta\lambda$
C12: $\delta\lambda$ [Å]: Spacing of wavelength points for spectrum synthesis
(C13: W_0 [mÅ]: total equivalent width of this blend, see below)
Row 6: Description for data in row 6
Row 7: `clam` and `gfscale` (see Sect.6.1)

In this case, $\Delta\overline{r^2}/a_0^2$ is computed from *LU*, *DIU*, *LO*, *DIO* (Function *rrca*). As before, the Stark broadening (due to collisions with electrons) is neglected ($C_4 = 0$). Radiative damping ($\gamma_{\rm rad}$) is treated in the classical approximation.

In the case of a **single blended line** the 'line.dat' file looks as follows:

Example:

```
Mult   namj   ei      alam     gflg     dlgC6  lu  diu  lo  dio  dlam   ddlam
1   3
O I ApJ Line 1: 67  6158.17 10.741 -1.140
3   1 1 1
  67   800    10.741  6158.15  -1.985   1.0    1   0.0  2   0.0
  67   800    10.741  6158.17  -1.140   1.0    1   0.0  2   0.0
  67   800    10.741  6158.19  -0.553   1.0    1   0.0  2   0.0  4.D-1  4.D-3
clam     gfscale
2000.0   1.0
```

Here $kline = 1$, $ktotal = 3$, $nbl = 3$, $incode = [1, 1, 1]$. Note that only the last of the rows describing the blend need entries $C11$ and $C12$.

As in the case of format '0', it is possible to enter an **equivalent width** (W_0 in [mÅ]) in column $C13$. For this purpose, *nbl* must be negative, with $|nbl|$ being the number of blend components. The gf value producing this equivalent width W_0 is returned in `result.gflg01` (average 3D atmosphere) and `result.gflg0x` (1D reference atmosphere).

Example unblended line:

```
Mult   namj  ei     alam     gflg   dlgC6 lu  diu  lo  dio  dlam  ddlam  W0
 1  1
```

```
O I ApJ Line 2: 92  6300.30  0.000 -9.773
-1  1
  92   800   0.000  6300.30  -9.773 1.0   1   0.0  2   0.0  4.D-1 4.D-3  7.00
clam     gfscale
2000.0   1.0
```

Example blended line:

```
Mult   namj   ei    alam     gflg   dlgC6 lu  diu  lo  dio  dlam  ddlam  W0
 1  3
O I ApJ Line 1: 67  6158.17 10.741 -1.140
-3  1 1 1
  67   800  10.741  6158.15  -1.985 1.0   1   0.0  2   0.0
  67   800  10.741  6158.17  -1.140 1.0   1   0.0  2   0.0
  67   800  10.741  6158.19  -0.553 1.0   1   0.0  2   0.0  4.D-1 4.D-3 10.00
clam     gfscale
2000.0   1.0
```

6.2.4 Single line calculations, complete line data format '2'

For a **single unblended line**, the this form of the 'line.dat' file looks like this.

Example:

```
Mult   namj   ei      alam     gflg   dlgC6  drrca1  dlgC4 C4lg   dlggr  Crad  dlam   ddlam
1   1
Si I AA Line 5: 16   5948.540  5.0823  -1.130  390.03   11.80  -1  86
1   2
   16  1400   5.0823  5948.540 -1.130  1.0   390.03  0.0   11.80  0.0    -1.0  5.D-1  5.D-3
clam     gfscale
2000.0   1.0
```

Description of entries:

Row 1: Header (identifies the meaning of the columns for data in row 5)
Row 2: Two integers, *kline* and *ktotal*
kline: number of line calculations requested in this file
ktotal: is the total number of spectral lines including blends
in this case $kline = 1$, $ktotal = 1$
Row 3: String, identifier of the (first) line calculation
Row 4: Integer *nbl*, integer array *incode*(*nbl*)
nb: number of blend components for this line calculation (= 1)
incode: integer array identifying the input format for each of the blend components (= 2)
Row 5: Line data in format '2' (11 + 2 columns):
C1: Multiplet number (for information only)
C2: Identifier of atom or ion (e.g. 2601 mean FeII)
C3: Excitation potential of lower level in [eV]
C4: Central wavelength of blend component
C5: $\log gf$ value of blend component
C6: $\Delta \log C_6$: Enhancement factor for van der Waals line broadening
C7: $\Delta\overline{r^2}/a_0^2$: Difference of mean square electron orbital radii
C8: $\Delta \log C_4$: Enhancement factor for Stark line broadening
C9: $-\log C_4$: Stark broadening constant.
if $-\log C_4 < 0$ (typically $-\log C_4 = -1.0$), then $C_4 = 0$
if $-\log C_4 = 0$, then use Griem (Phys. Rev. 165, 258, 1968)

and Cowley (Obs. 91, 139, 1971) approximation
C10: $\Delta \log \gamma_{\rm rad}$: Enhancement factor for natural line broadening
C11: $C_{\rm rad}$: Natural line broadening ($10^{-8}\gamma_{\rm rad}$)
if $C_{\rm rad} < 0$, use classical formula ($\gamma_{\rm rad} = 2.22 \cdot 10^{15}/\lambda^2$) [1/s], where λ is in Å.
C12: $\Delta\lambda$ [Å]: Line profile is computed from $\lambda_0 - \Delta\lambda$ to $\lambda_0 + \Delta\lambda$
C13: $\delta\lambda$ [Å]: Spacing of wavelength points for spectrum synthesis
(C14: W_0 [mÅ]: total equivalent width of this blend, see below)
Row 6: Description for data in row 6
Row 7: `clam` and `gfscale` (see Sect.6.1)

In the case of a **single blended line** the 'line.dat' file looks as follows:

Example:

```
Mult   namj   ei      alam     gflg   dlgC6  drrca1  dlgC4 C4lg   dlggr  Crad  dlam   ddlam
1   2
Si I / Si II blend: 16   5948.540  5.0823  -1.130  390.03   11.80  -1  86
2   2 2
   16  1400   5.0823  5948.540 -1.130  1.0   390.03  0.0   11.80  0.0    -1.0
   16  1401   0.0823  5948.530 -3.130  1.0    90.00  0.0   13.80  0.0    -1.0  5.D-1  5.D-3
clam     gfscale
2000.0   1.0
```

Here $kline = 1$, $ktotal = 2$, $nbl = 2$, $incode = [2, 2, 2]$. Note that only the last of the rows describing the blend need entries $C12$ and $C13$.

As in the cases of format '0' and '1', it is possible to enter an **equivalent width** (W_0 in [mÅ]) in column $C14$. For this purpose, nbl must be negative, with $|nbl|$ being the number of blend components. The gf value producing this equivalent width W_0 is returned in `result.gflg01` (average 3D atmosphere) and `result.gflg0x` (1D reference atmosphere). No examples are given since the necessary modification the the data format should be obvious.

6.2.5 Single line calculations, complete line data format '3'

This data format has a maximum of 17 columns. It differs from format '2' only in the way the van der Waals broadening parameters are specified. Columns $C7$ with $\Delta\overline{r^2}/a_0^2$ is replaced by the four columns:

C7: LU: Orbital quantum number of valence electron of lower level
C8: DIU: excitation energy $[eV]$ of parent term for lower level
C9: LO: Orbital quantum number of valence electron of upper level
C10: DIO: excitation energy $[eV]$ of parent term for upper level

as in format '1'. The remaining columns are as in format '2', but shifted by +3:

C11: $\Delta \log C_4$: Enhancement factor for Stark line broadening
C12: $-\log C_4$: Stark broadening constant.
if $-\log C_4 < 0$ (typically $-\log C_4 = -1.0$), then $C_4 = 0$
if $\log C_4 = 0$, then use Griem (Phys. Rev. 165, 258, 1968)
and Cowley (Obs. 91, 139, 1971) approximation
C13: $\Delta \log \gamma_{\rm rad}$: Enhancement factor for natural line broadening
C14: $C_{\rm rad}$: Natural line broadening ($10^{-8}\gamma_{\rm rad}$)
if $C_{\rm rad} < 0$, use classical formula ($\gamma_{\rm rad} = 2.22 \cdot 10^{15}/\lambda^2$) [1/s], where λ is in Å.
C15: $\Delta\lambda$ [Å]: Line profile is computed from $\lambda_0 - \Delta\lambda$ to $\lambda_0 + \Delta\lambda$

C16: $\delta\lambda$ [Å]: Spacing of wavelength points for spectrum synthesis
(C17: W_0 [mÅ]: total equivalent width of this blend, see below)

This format has not yet been used or tested, and no example files are available.

As in the cases of format '0', '1', and '2' it should also be possible to enter an **equivalent width** (W_0 in [mÅ]), now in column $C17$. For this purpose, *nbl* must be negative, with $|nbl|$ being the number of blend components. The gf value producing this equivalent width W_0 is returned in `result.gflg01` (average 3D atmosphere) and `result.gflg0x` (1D reference atmosphere).

6.2.6 Single line calculations, complete line data format '7'

This data format was designed for simple test calculations where the line profile is fixed, i.e. the line parameters are depth-independent (see also Sect. 3.5). This format has a maximum of 7 columns:

Description of entries:

Row 1: Header (identifies the meaning of the columns for data in row 5)
Row 2: Two integers, *kline* and *ktotal*
kline: number of line calculations requested in this file
ktotal: is the total number of spectral lines including blends
in this case $kline = 1$, $ktotal = 1$
Row 3: String, identifier of the line calculation
Row 4: Integer *nbl*, integer array *incode*(*nbl*)
nb: number of blend components for this line calculation (= 1)
incode: integer array identifying the input format for each of the blend components (= 7)
Row 5: C1: Central wavelength of blend component [Å]
C2: Doppler broadening in units of c, $v_{\rm D}/c$
C3: $\eta_0 = \kappa_{\rm line}/\kappa_{\rm cont}$ at line center
C4: α-parameter for Voigt profile, $\alpha = \gamma/2/\Delta\omega_{\rm D}$
(ratio of half half width of dispersion profile and Doppler widths of Gaussian).
C5: $\Delta\lambda$ [Å]: Line profile is computed from $\lambda_0 - \Delta\lambda$ to $\lambda_0 + \Delta\lambda$
C6: $\delta\lambda$ [Å]: Spacing of wavelength points for spectrum synthesis
(C7: W_0 [mÅ]: total equivalent width of this blend, see above)
Row 6: Description for data in row 6
Row 7: `clam` and `gfscale` (see Sect.6.1)

Example:

```
alam     Vdop     eta0    avgt    dlam    ddlam
1    1
Test    grey sf  Vdop=2.D-5, eta0=1.0D0, avgt=1.D-2
1   7
4000.000 2.0D-5   1.0D0   1.0D-2  0.90D0  0.90D-2
clam        gfscale
-4000.000   1.0
```

6.2.7 Multiple Line Calculations

It is also possible to process a whole set of lines in a single run. The requirement is, however, that all lines have the same central wavelength (continuum wavelength). This mode was designed for parameter studies, e.g. investigating the "granulation abundance corrections" as a function of line excitation potential.

Example, 8 unblended N I lines of different excitation potential:

```
Mult   namj  chik    alam      gflg    dlgC6  drrca1  dlam     ddlam      W0
  8  8
  N I Fictitious Line  1: /   0.000   5500.0  -7.6914  1.00   10.00   75.00 /
 -1  0
9999   700   0.000   5500.0  -7.6914  1.00   10.00  3.00E-01  3.00E-03   75.00
  N I Fictitious Line  2: /   2.000   5500.0  -5.7282  1.00   10.00   75.00 /
 -1  0
9999   700   2.000   5500.0  -5.7282  1.00   10.00  3.00E-01  3.00E-03   75.00
  N I Fictitious Line  3: /   4.000   5500.0  -3.8298  1.00   10.00   75.00 /
 -1  0
9999   700   4.000   5500.0  -3.8298  1.00   10.00  3.00E-01  3.00E-03   75.00
  N I Fictitious Line  4: /   6.000   5500.0  -1.9876  1.00   10.00   75.00 /
 -1  0
9999   700   6.000   5500.0  -1.9876  1.00   10.00  3.00E-01  3.00E-03   75.00
  N I Fictitious Line  5: /   8.000   5500.0  -0.1961  1.00   10.00   75.00 /
 -1  0
9999   700   8.000   5500.0  -0.1961  1.00   10.00  3.00E-01  3.00E-03   75.00
  N I Fictitious Line  6: /  10.000   5500.0   1.5485  1.00   10.00   75.00 /
 -1  0
9999   700  10.000   5500.0   1.5485  1.00   10.00  3.00E-01  3.00E-03   75.00
  N I Fictitious Line  7: /  11.000   5500.0   2.4046  1.00   10.00   75.00 /
 -1  0
9999   700  11.000   5500.0   2.4046  1.00   10.00  3.00E-01  3.00E-03   75.00
  N I Fictitious Line  8: /  12.000   5500.0   3.2510  1.00   10.00   75.00 /
 -1  0
9999   700  12.000   5500.0   3.2510  1.00   10.00  3.00E-01  3.00E-03   75.00
clam     gfscale
2000.0   1.0
```

Note that now $kline = 8$, and $ktotal = 8$, since all lines have one blend component only.

6.3 Conversion of line broadening parameters

The line broadening can be specified in different ways, e.g. as $-\log C_4$ for quadratic Stark broadening. The required data is, however, not always available and must be converted from other broadening parameters, e.g. γ_4. In the particular case of the *Vienna Atomic Line Database* the broadening is provided as $\log(\gamma_4/N_e)$ for a temperature of $T = 10^4\,\mathrm{K}$.

Please note that here and in Linfor3D in general, the parameters C_n $(n = 4, 6)$ are defined via

$$\Delta\omega = \frac{C_n}{r^n} \tag{79}$$

whereas the definition by Unsöld is

$$\Delta\omega = 2\,\pi\,\frac{C_n}{r^n}. \tag{80}$$

The Linfor parameters C_n are thus a factor 2π larger than in the definition by Unsöld.

6.3.1 Quadratic Stark effect

The broadening parameter γ_4 for the quadratic Stark effect can be written as

$$\gamma_4 = 11.37\,C_4^{2/3}\,{v_{\mathrm{rel}}}^{1/3}\,N_{\mathrm{e}}\ , \tag{81}$$

where v_{rel} is the relative velocity between the regarded atom and the perturber, i.e. the colliding particle:

$${v_{\mathrm{rel}}}^2 = \frac{8\,k\,T}{\pi\,m_{\mathrm{H}}}\cdot\left(\frac{1}{A_1}+\frac{1}{A_2}\right)\ . \tag{82}$$

A_1 and A_2 are the atomic weights in atomic mass units, e.g., $A_2 = 1$ for a colliding hydrogen atom and $A_1 \approx 56$ for iron atoms and $A_2 = 1/1837 = m_\mathrm{e}/m_\mathrm{H}$ for electrons. For Stark broadening with electrons as perturbers the following good approximation can be made:

$$A_1 >> A_2 \Rightarrow \frac{1}{A_1} + \frac{1}{A_2} \approx \frac{1}{A_2} = 1837 = m_\mathrm{H}/m_\mathrm{e} \tag{83}$$

With this Eq. 81 can be written as

$$\begin{aligned}
\log \frac{\gamma_4}{N_\mathrm{e}} &= \log 11.37 + \log C_4^{2/3} + \log {v_\mathrm{rel}}^{1/3} && (84)\\
&= 1.056 + \frac{2}{3}\log C_4 + \frac{1}{6}\log \frac{8\,k\,T}{\pi\, m_\mathrm{e}} && (85)\\
&= 1.056 + \frac{2}{3}\log C_4 + 1.931 + \frac{1}{6}\log T && (86)\\
&= 1.056 + \frac{2}{3}\log C_4 + 1.931 + \frac{1}{6}\log 10^4 + \frac{1}{6}\log \frac{T}{10^4\,\mathrm{K}} && (87)\\
& && (88)
\end{aligned}$$

With $T = 10^4$ K, which is assumed for data in VALD, we derive

$$\log \frac{\gamma_4}{N_\mathrm{e}} = 3.654 + \frac{2}{3}\log C_4 \tag{89}$$

and finally the conversion formula:

$$\log C_4 = 1.5 \log \frac{\gamma_4}{N_\mathrm{e}} - 5.4805 \tag{90}$$

For instance a value of -5.491 from VALD gives $\log C_4 = -13.717$. The parameter `C4lg` is thus set to `13.717`.

6.3.2 Van der Waals broadening

The broadening parameter γ_6 for the van der Waals effect can be written as

$$\gamma_6 = 8.08\, {C_6}^{2/5}\, {v_\mathrm{rel}}^{3/5}\, N_\mathrm{H} \ . \tag{91}$$

The perturbing particles are mostly hydrogen atoms with $A_2 = 1$. We now make the approximation

$$A_1 > A_2 \Rightarrow \frac{1}{A_1} + \frac{1}{A_2} \approx \frac{1}{A_2} = 1 \tag{92}$$

With this the relative velocity of the particles (Eq. 82) reduces to

$${v_\mathrm{rel}}^2 = \frac{8\,k\,T}{\pi\, m_\mathrm{H}}. \tag{93}$$

We can thus rewrite Eq. 91:

$$\begin{aligned}
\log \frac{\gamma_6}{N_\mathrm{H}} &= \log 8.08 + \log C_6^{2/5} + \log {v_\mathrm{rel}}^{3/5} && (94)\\
&= 0.907 + \frac{2}{5}\log C_6 + \frac{3}{10}\log \frac{8\,k\,T}{\pi\, m_\mathrm{H}} && (95)\\
&= 0.907 + \frac{2}{5}\log C_6 + 2.497 + \frac{3}{10}\log T && (96)\\
&= 0.907 + \frac{2}{5}\log C_6 + 2.497 + \frac{3}{10}\log 10^4 + \frac{3}{10}\log \frac{T}{10^4\,\mathrm{K}} && (97)\\
& && (98)
\end{aligned}$$

With $T = 10^4$ K, which is assumed for data in VALD, we derive

$$\log \frac{\gamma_6}{N_\mathrm{H}} = 4.604 + \frac{2}{5} \log C_6 \tag{99}$$

and finally the conversion formula:

$$\log C_6 = 2.5 \log \frac{\gamma_6}{N_\mathrm{H}} - 11.510 \tag{100}$$

For instance a value of -7.619 from VALD gives $\log C_6 = -30.558$. Unfortunately there is no parameter `C6lg` which could be set directly in the present code version. Instead the van der Waals broadening can be specified via the difference of mean square electron orbital radii $\Delta\overline{r^2}/a_0^2$ where a_0 is the Bohr radius:

$$\log \left(\Delta\overline{r^2}/a_0^2 \right) = \log C_6 + 32.3867 \ . \tag{101}$$

We finally derive a conversion formula:

$$\Delta\overline{r^2}/a_0^2 = 10^{20.877 + 2.5 \log \frac{\gamma_6}{N_\mathrm{H}}} \ . \tag{102}$$

The exemplary value of -7.619 from VALD thus gives `67.437` for the parameter `drrca1`. In addition `dlgC6` should be set to 0 unless you want to apply an additional enhancement of the broadening.

6.3.3 Natural line broadening

The broadening parameter γ_rad can be converted like this:

$$C_\mathrm{rad} = 10^{\log \gamma_\mathrm{rad} - 8.0} \tag{103}$$

For instance, $\log \gamma_\mathrm{rad} = 7.877$ would give $C_\mathrm{rad} = 0.753$. The parameter `Crad` is thus set to `0.753`, and `dlggr` is set to `0.0`.

7 Output files

LINFOR3D generates the following output files in the LINFOR3D directory:

name		content
`linfor_3D_1.idlsave`	:	IDL structures `cmd`,`const`,`line`,`result`
`linfor_3D_2.idlsave`	:	IDL structure `maps` (see below for more details)
`linfor_timing.txt`	:	Timing statistics (see Sect. 8)
`linfor_3D_1.ps`	:	Postscript file: local line profiles plus average
`linfor_3D_2.ps`	:	Postscript file: line profiles for 1D reference atmosphere and time-averaged 1D and 3D spectra; granulation abundance correction

7.1 Files containing intensity maps

The output file `linfor_3D_2.idlsave` contains a structure `MAPS`. An example of this structure is:

```
** Structure <83027f4>, 11 tags, length=11289820, data length=11289820, refs=1:
   NX              LONG                 140
   NY              LONG                 140
   NDATA           INT             12
   KLINE           INT              1
   NLAM            LONG                 11
   MODELID         STRING    Array[12]
   MU0             FLOAT           1.00000
   PHI0            FLOAT           0.00000
   CLAM            FLOAT           3966.34
   LINEID          STRING    Array[1]
   DV3             FLOAT     Array[11]
   ICLAM0          FLOAT     Array[140, 140, 12]
   ICLAM2          FLOAT     Array[140, 140, 11, 1, 12]
```

Depending on the value of the control parameter `maps_flag` (see Sect. 5.1), there might be a tag named `ICLAM1`

```
   ICLAM1          FLOAT     Array[140, 140, 12, 1]
```

instead of `ICLAM2` or even both might be missing if `maps_flag` = 0.

Description of entries:

`nx`, `ny`	:	x,y dimensions of the 2D images
`ndata`	:	number of models for which spectrum synthesis was done
`kline`	:	number of lines for which spectrum synthesis was done
`clam`	:	central continuum wavelength for all maps
`modelid`	:	model identifier (0:ndata-1)
`lineid`	:	line identifier (0:kline-1)
`ICLAM0`	:	continuum intensity maps for all models (at clam), dimensions: `nx`, `ny` (see Sect. 6)
`ICLAM1`	:	emergent intensity maps for all models and lines, including line absorption at clam (window center); dimensions: `nx`, `ny`, `ndata`, `kline`; only present if `maps_flag` = 1 (see Sect. 6)
`ICLAM2`	:	emergent intensity maps for all models and lines, including line absorption at all wavelengths within the wavelength window of width 2 · `dlam` around the central wavelength `clam`: λ_i = `clam` − `dlam` + i · `ddlam`, where `clam` = `alam` + `dclam`, `alam` is the wavelength of the main blend component as defined in `line.dat`; dimensions: `nx`, `ny`, `nlam`, `kline`, `ndata`; only present if `maps_flag` = 2 (see Sect. 6)
`W3LAM`	:	equivalent width maps for all models and lines

Note:

- Intensities are given in units of [erg cm^{-2} s^{-1} sr^{-1} Å^{-1}].
- The file formerly called "linfor_3D.idlsave" was renamed to "linfor_3D_1.idlsave".

7.2 Foreshortening

Note that the maps include foreshortening effects. A model with a quadratic cross section becomes a rectangle when viewed off-center.

If `ntheta`≠ 0, the flux spectrum is computed as before, and the intensity maps show the vertical view, as before.

Keyword `view` added to plotting routine `linfor_plot3`. If given, the intensity and equivalent width maps show the foreshortened view.

8 Timing statistics

At the end of a run of LINFOR3D timing statistics are presented which are also saved to the file `linfor_timing.txt` in the current directory. This file could look like this:

```
-------------------------------------------------------------------------------
  T I M I N G   S T A T I S T I C S
-------------------------------------------------------------------------------
  Routine linfor_find_ff.....(total   ) :         0.02 s  (   0.30 % )
  Restoring structure FF.....(total   ) :         0.08 s  (   1.55 % )
                             (average ) :         0.01 s  (   0.26 % )
                             (       0) :         0.03 s  (   0.55 % )
                             (       1) :         0.01 s  (   0.19 % )
                             (       2) :         0.01 s  (   0.19 % )
                             (       3) :         0.01 s  (   0.19 % )
                             (       4) :         0.01 s  (   0.19 % )
                             (       5) :         0.01 s  (   0.25 % )
  Reading 3D model..............     :          ---
  Routine linfor_ionopa_3d........   :          ---
  Saving structure FF.............   :          ---
  Radiative transfer for 3D model.(total   ) :         2.08 s  (  39.07 % )
                             (average ) :         0.35 s  (   6.51 % )
                             (       0) :         0.34 s  (   6.41 % )
                             (       1) :         0.34 s  (   6.44 % )
                             (       2) :         0.35 s  (   6.54 % )
                             (       3) :         0.36 s  (   6.68 % )
                             (       4) :         0.35 s  (   6.51 % )
                             (       5) :         0.35 s  (   6.49 % )
  Total..........................(total   ) :         5.33 s  ( 100.00 % )
-------------------------------------------------------------------------------
```

The file is also saved during a running LINFOR3D process. Thus, time statistics are available even after aborting the process. The statistics show the system time needed for individual computation steps/routines of LINFOR3D and their contribution to the total time in percent. For the case that the same operation is performed several times, e.g., doing the radiative transfer for more than one model snap shot, the total of all calls, the average time, and the duration for each individual step is given (see example above).

9 IONDIS

9.1 Molecules

Table 4: Small molecular network: 5 atoms, 8 molecules

	H	C	N	O	Mg
H	H_2	CH	NH	OH	MgH
C	CH	C_2	CN	CO	—
N	NH	CN	–	–	—
O	OH	CO	–	–	—
Mg	MgH	–	–	–	—

Table 5: Large molecular network:10 atoms, 14 molecules

	H	Li	C	N	O	F	Mg	Ti	Cr	Fe
H	H_2	LiH	CH	NH	OH	FH	MgH	—	CrH	FeH
Li	LiH	—	–	–	LiO	–	—	—	—	—
C	CH	—	C_2	CN	CO	–	—	—	—	—
N	NH	—	CN	–	–	–	—	—	—	—
O	OH	LiO	CO	–	–	–	—	TiO	—	—
F	FH	—	–	–	–	–	—	—	—	—
Mg	MgH	—	–	–	–	–	—	—	—	—
Ti	—	—	–	–	TiO	–	—	—	—	—
Cr	CrH	—	–	–	–	–	—	—	—	—
Fe	FeH	—	–	–	–	–	—	—	—	—

9.1.1 Some definitions

N_i	Total number density of nuclei of element i, including nuclei bound in diatomic molecules
$\tilde{N}_i$	Number density of nuclei of element i not bound in diatomic molecules
$\tilde{N}_{i,0}$	Number density of *neutral* nuclei of element i not bound in diatomic molecules
$N_{i,k}$	Total number density of diatomic molecules made up of one nucleus of element i, and one nucleus of element k
N_e	Total electron number density.
$N_{\rm Kern} = \sum_i N_i$	Total number density of nuclei of all elements, diatomic molecules counting as 2 nuclei
$P_e = N_e\, kT$	Elektron pressure.
$f_i = N_i/N_{\rm Kern}$	Fractional abundance of element i (constant)
$x_i = \tilde{N}_i/N_{\rm Kern}$	Fractional abundance of free nuclei of element i. $x_i = f_i = const.$ for elements not involved in molecule formation ($i \notin \{i_{mol}\}$). For elements forming molecules ($i \in \{i_{mol}\}$), x_i is the variable to be iterated.
$x_{i,k} = N_{i,k}/N_{\rm Kern}$	Fractional abundance of molecule (i,k).
$x_e = N_e/N_{\rm Kern}$	Fractional abundance of free electrons (iterated quantity).

9.1.2 Equations

The Saha equation provides the relation between $\tilde{N}_{i,0}$ and $\tilde{N}_i$:

$$\tilde{N}_{i,0} = S_{i,0}\, \tilde{N}_i \tag{104}$$

where the Saha factor $S_{i,0}$ depends on temperature and electron pressure (and on the ionization potentials and the partition functions of the different ionization stages).

Molecule partial pressures are given by the relation

$$P_{i,k} = \frac{P_i \, P_k}{K_{i,k}} \tag{105}$$

where $K_{i,k}$ is the dissociation constant for the (neutral) diatomic molecule (ik), composed of one nucleus of elements i and k each. P_i and P_k are the partial pressures of the **neutral** atoms of elements i and k, respectively. Since $P = N\,kT$, molecule densities are

$$N_{i,k} = \frac{kT \, \tilde{N}_{i,0} \, \tilde{N}_{k,0}}{K_{i,k}} \tag{106}$$

Dividing by $N_{\rm Kern}$, we obtain the fractional molecule abundance

$$x_{i,k} = \frac{N_{i,k}}{N_{\rm Kern}} = \frac{N_{\rm Kern}\, kT \, x_i S_{i,0} \, x_k S_{k,0}}{K_{i,k}} = \frac{P_e \, x_i S_{i,0} \, x_k S_{k,0}}{K_{i,k} \, x_e} = D_{i,k} \, \frac{x_i \, x_k}{x_e} \tag{107}$$

where we have defined

$$D_{i,k} = P_e \, \frac{S_{i,0} \, S_{k,0}}{K_{i,k}}. \tag{108}$$

We note that $D_{i,k}$ depends only on T and P_e, but not on absolute number densities, and hence is constant during the iteration.
For all elements i involved in molecule formation, $i \in \{i_{mol}\}$, we have the following conservation equation

$$f_i = x_i + \sum_k x_{i,k} \, (1 + \delta_{i,k}) \tag{109}$$

or

$$x_i + \sum_k D_{i,k} \, \frac{x_i \, x_k}{x_e} \, (1 + \delta_{i,k}) = f_i \tag{110}$$

where $\delta_{i,k}$ is the Kronecker symbol, accounting for the correct counting of atoms in molecules with two identical components.
The electron density is given by

$$N_e = \sum_i \tilde{N}_i \, \overline{Z}_i = \sum_{i \in \{i_{mol}\}} \tilde{N}_i \, \overline{Z}_i \; + \sum_{i \notin \{i_{mol}\}} \tilde{N}_i \, \overline{Z}_i \tag{111}$$

where the mean degree of ionization of element i, $\overline{Z}_i$ is defined as

$$\overline{Z}_i = \sum_{j=-1,3} j \, \tilde{N}_{i,j} \, / \sum_{j=-1,3} \tilde{N}_{i,j} = \frac{1}{\tilde{N}_i} \sum_{j=-1,3} j \, \tilde{N}_{i,j} = \sum_{j=-1,3} j \, \tilde{S}_{i,j} = \sum_{j=1,3} j \, \tilde{S}_{i,j} - \tilde{S}_{i,-1}. \tag{112}$$

Here index j runs over the 4 ionization stages $j = 0 \ldots 3$ of element i. For elements forming negative ions, the sum includes the negative ion, $j = -1$. For such elements, $\overline{Z}_i$ may become negative! Note that $\overline{Z}_i$ depends only on T and P_e, but not on the degree of molecule formation. Dividing Eq.(111) by $N_{\rm Kern}$, we obtain

$$x_e = \sum_i x_i \, \overline{Z}_i = \sum_{i \in \{i_{mol}\}} x_i \, \overline{Z}_i \; + \sum_{i \notin \{i_{mol}\}} x_i \, \overline{Z}_i \tag{113}$$

Defining

$$f_e = \sum_{i \notin \{i_{mol}\}} x_i \, \overline{Z}_i = \sum_{i \notin \{i_{mol}\}} f_i \, \overline{Z}_i = const. \tag{114}$$

we have finally

$$x_e - \sum_{i \in \{i_{mol}\}} x_i \, \overline{Z}_i = f_e \tag{115}$$

Combining Eq.(110) and (115), we have the following vector equation

$$\vec{X} + \vec{F}(\vec{X}) = \vec{R} \tag{116}$$

where

$$\vec{X} = \{x_1, x_2, \ldots x_N, x_e\}, \tag{117}$$

is the vector of unknown number fractions, and

$$\vec{R} = \{f_1, f_2, \ldots f_N, f_e\} = const., \tag{118}$$

is the known (constant) right-hand side. N is the number of chemical elements included in the molecular network. Equation (116) is a system of $N+1$ nonlinear algebraic equations which can be solved for $\vec{X}$ by Newton-Raphson iteration.
The first step it to find a suitable starting vector for the iteration, $\vec{X}_0$. This is done as described below. The correction $\delta\vec{X}$ giving the next improved estimate of $\vec{X}$ is computed as follows. Assume after n iterations we have

$$\vec{X}_n + \vec{F}(\vec{X}_n) = \vec{R}_n. \tag{119}$$

Then we require that

$$\vec{X}_n + \delta\vec{X} + \vec{F}(\vec{X}_n + \delta\vec{X}) = \vec{R} \tag{120}$$

or

$$\vec{X}_n + \delta\vec{X} + \vec{F}(\vec{X}_n) + \mathcal{J} \cdot \delta\vec{X} = \vec{R}, \tag{121}$$

hence

$$(\mathcal{J} + 1) \cdot \delta\vec{X} = \vec{R} - \vec{R_n}. \tag{122}$$

The elements of the Jacobian $\mathcal{J}$ are defined as

$$\mathcal{J}_{i,j} = \frac{\partial F_i}{\partial x_j}. \tag{123}$$

Since we know that

$$F_i(\vec{X}) = \frac{x_i}{x_e} \sum_{k=1,N} D_{i,k} \, x_k \, (1 + \delta_{i,k}) \quad \text{for } i = 1, N \tag{124}$$

and

$$F_{N+1}(\vec{X}) = - \sum_{i=1,N} x_i \, \overline{Z}_i, \tag{125}$$

we can readily evaluate $\mathcal{J}_{i,j}$.
We find from Eqs.(124) and (125)

$$\mathcal{J}_{i,j} = \frac{\partial F_i}{\partial x_j} = D_{i,j} \, \frac{x_i}{x_e} \quad \text{for } i = 1, N \text{ and } i \neq j \tag{126}$$

$$\mathcal{J}_{i,i} = \frac{\partial F_i}{\partial x_i} = \sum_{k \neq i} D_{i,k} \, \frac{x_k}{x_e} + 4 \, D_{i,i} \, \frac{x_i}{x_e} = \sum_{k=1,N} D_{i,k} \, \frac{x_k}{x_e} + 3 \, D_{i,i} \, \frac{x_i}{x_e} \quad \text{for } i = 1, N \tag{127}$$

$$\mathcal{J}_{i,N+1} = \frac{\partial F_i}{\partial x_e} = -\frac{x_i}{x_e^2} \sum_{k=1,N} D_{i,k} \, x_k \, (1 + \delta_{i,k}) \quad \text{for } i = 1, N \tag{128}$$

$$\mathcal{J}_{N+1,j} = \frac{\partial F_{N+1}}{\partial x_j} = -\overline{Z}_i \quad \text{for } j = 1, N \tag{129}$$

$$\mathcal{J}_{N+1,N+1} = \frac{\partial F_{N+1}}{\partial x_e} = 0 \tag{130}$$

With this information , we can solve Eq.(122) for $\delta\vec{X}$, and obtain the next estimate

$$\vec{X}_{n+1} = \vec{X}_n + \delta\vec{X} \tag{131}$$

Once the iteration has converged, the molecule densities can be computed from Eq.(107).

9.1.3 Criterion for convergence

The criterion for convergence is currently:

$$|x_i^{(n+1)} - x_i^{(n)}| \leq 1 \cdot 10^{-4}\, f_i \tag{132}$$

and

$$|x_i^{(n)} + \sum_k D_{i,k}\, \frac{x_i^{(n)}\, x_k^{(n)}}{x_e^{(n)}}\, (1 + \delta_{i,k}) - f_i| \leq 1 \cdot 10^{-4}\, f_i \tag{133}$$

for all elements i. The maximum number of iterations is 15.

9.1.4 Initial guess

The initial concentrations of free atoms and ions of elements involved in molecule formation, x_i, are computed as follows.
First, we assume that no molecules are formed and so the initial x_i are set to f_i,

$$x_{i,0} = f_i \tag{134}$$

for all elements. From this, the electron fraction x_e is computed as

$$x_{e,0} = \max\left\{ x_{e,\min},\ f_e + \sum_{i \in \{i_{mol}\}} x_i\, \overline{Z}_i \right\}, \tag{135}$$

where $x_{e,\min} = 1 \cdot 10^{-10}$. Using this value for x_e, we compute the molecule concentrations $x_{i,k}$ according to Eq.(107). If the resulting

$$x_{i,k} \leq 1 \cdot 10^{-5}\, \min\{x_i, x_k\}, \tag{136}$$

the formation of this molecule is considered negligible, and no correction of x_i, x_k and $x_{i,k}$ is necessary. If

$$1 \cdot 10^{-5} \min\{x_i, x_k\} < x_{i,k} \leq \min\{x_i, x_k\}, \tag{137}$$

molecule formation is no longer negligible, but also not exhaustive. In this case, the molecule concentrations must be iterated, but the initial guesses for $x_{i,k}$, x_i and x_k need not be changed. Finally, if

$$x_{i,k} > \min\{x_i, x_k\}, \tag{138}$$

then molecule formation is exhaustive, and the initial guesses for $x_{i,k}$, x_i and x_k are changed. We compute x_i and x_k as the equilibrium values that would result if only this particular molecule was present.
If the molecule consists of two atoms of the same element, the condition is (see Eq.(110))

$$2\, D_{i,i}\, \frac{x_i^2}{x_e} + x_i - f_i = 0 \tag{139}$$

which has the solution

$$x_{i,0} = \frac{2\,f_i}{1+\sqrt{1+8\,f_i\,D_{i,i}/x_{e,0}}}. \tag{140}$$

If the molecule consists of two different atoms, we have two conditions, namely (see Eq.(110))

$$\begin{aligned} D_{i,k}\,\frac{x_i\,x_k}{x_e} + x_i - f_i &= 0 \\ D_{i,k}\,\frac{x_i\,x_k}{x_e} + x_k - f_k &= 0 \end{aligned} \tag{141}$$

From this we see that

$$x_i - x_k = f_i - f_k \equiv \Delta. \tag{142}$$

Then we have

$$D_{i,k}\,\frac{(x_k+\Delta)\,x_k}{x_e} + x_k - f_k = 0 \tag{143}$$

or

$$\frac{D_{i,k}}{x_e}\,x_k^2 + \left(1+\Delta\,\frac{D_{i,k}}{x_e}\right)\,x_k - f_k = 0. \tag{144}$$

Assuming that $f_i \geq f_k$, and hence $\Delta \geq 0$, the solution for x_k can be written as

$$x_{k,0} = \frac{2\,f_k}{(1+\Delta\,D_{i,k}/x_{e,0}) + \sqrt{(1+\Delta\,D_{i,k}/x_{e,0})^2 + 4\,f_i\,D_{i,k}/x_{e,0}}}, \tag{145}$$

and for x_i we simply have

$$x_{i,0} = x_{k,0} + \Delta. \tag{146}$$

For each molecule, the values of $x_{i,0}$, $x_{k,0}$ obtained for the current molecule from Eq.(140) or Eqns.(145), (146) are compared to the previous values $x_{i,0}^{(n)}$, $x_{k,0}^{(n)}$ (obtained from the same conditions for a previous molecule). The new estimate of is then set to the minimum of previous and current estimate

$$\begin{aligned} x_{i,0}^{(n+1)} &= \min\left\{x_{i,0},\; x_{i,0}^{(n)}\right\} \\ x_{k,0}^{(n+1)} &= \min\left\{x_{k,0},\; x_{k,0}^{(n)}\right\} \end{aligned} \tag{147}$$

The initial guess for the current molecule is then computed as

$$x_{i,k} = D_{i,k}\,\frac{x_{i,0}^{(n+1)}\;x_{k,0}^{(n+1)}}{x_{e,0}}. \tag{148}$$

For simplicity (and stability), the initial guess for the electron fraction is not updated when changing the initial guesses x_i for the elements involved in molecule formation.

9.1.5 Variable names

`DAB`	$D_{i,k}/x_e$	
`DFOO`	$\vec{R}-\vec{R}_n$	
`FRACEL`	x_e	
`FRACEL0`	f_e	
`FRACEL1`	$\sum_{i\in\{i_{mol}\}} f_i\,\overline{Z}_i$	
`FRACI`	f_i	
`FRACJ`	$x_i\,S_{i,j}$	(atoms and ions, $j=1\ldots4$)
	$x_{i,k}$	(molecules)
`FREEI`	x_i	
`NATMOL`	N	

PNUC	$N_{\mathrm{Kern}}\, k\, T = P_e / x_e$	
RIJSUM	F_i	
SAHAJ	$S_{i,j}$	(atoms and ions, $j = 1 \ldots 4$)
	$K_{i,k}$	(molecules)
ZEFF	$\overline{Z}_i$	

Index

Contribution functions, 10

Grey test case, 12

IDL, 18
IONDIS, 51

Line Data
    line.dat, 38
LINFOR3D, 6

Main program
    Calling sequence, 17

Parameter input
    linfor_setcmd, 20

Radiative transfer, 7

Structures, 18
structures
    flow field, 19
    ray system, 19
    spectrum, 19

Transfer equation for the continuum intensity, 7
Transfer equation for the line depression, 9
Transfer equation for the line intensity, 8